HOLTEK

I/O 8-Bit Flash MCU with LED/LCD Driver

HT69F240

Revision: V1.20 Date: March 25, 2015

www.holtek.com

Table of Contents

Features

CPU Features

- Operating voltage:
  - f_{SYS}=4MHz: 2.2V~5.5V
  - f_{SYS}=8MHz: 2.2V~5.5V
  - f_{SYS}=12MHz: 2.7V~5.5V
  - f_{SYS}=16MHz: 4.5V~5.5V
- Up to 0.25μs instruction cycle with16MHz system clock at V_{DD}=5V
- Power down and wake-up functions to reduce power consumption
- Oscillator types:
  - External Crystal – HXT
  - External 32.765kHz Crystal – LXT
  - Internal RC – HIRC
  - Internal 32kHz RC – LIRC
- Multi-mode operation: NORMAL, SLOW, IDLE and SLEEP
- Fully integrated internal 4MHz, 8MHz and 12MHz oscillator requires no external components
- All instructions executed in one or two instruction cycles
- Table read instructions
- 63 powerful instructions
- Up to 8-level subroutine nesting
- Bit manipulation instruction

Peripheral Features

- Flash Program Memory: 4K×16
- Data Memory: 256×8
- EEPROM Memory: 64×8
- Watchdog Timer function
- Up to 26 bidirectional I/O lines
- 24 bidirectional I/O lines with LED direct driving capability
- Software LCD driver
- I^2C Interface
- Universal Asynchronous Receive Transmit (UART) Interface
- Multiple pin-shared external interrupts
- Multiple Timer Module for time measure, input capture, compare match output, PWM output or single pulse output function
- Dual Time-Base functions for generation of fixed time interrupt signals
- Low voltage reset function
- Low voltage detect function
- Package types: 24/28-pin SOP

General Description

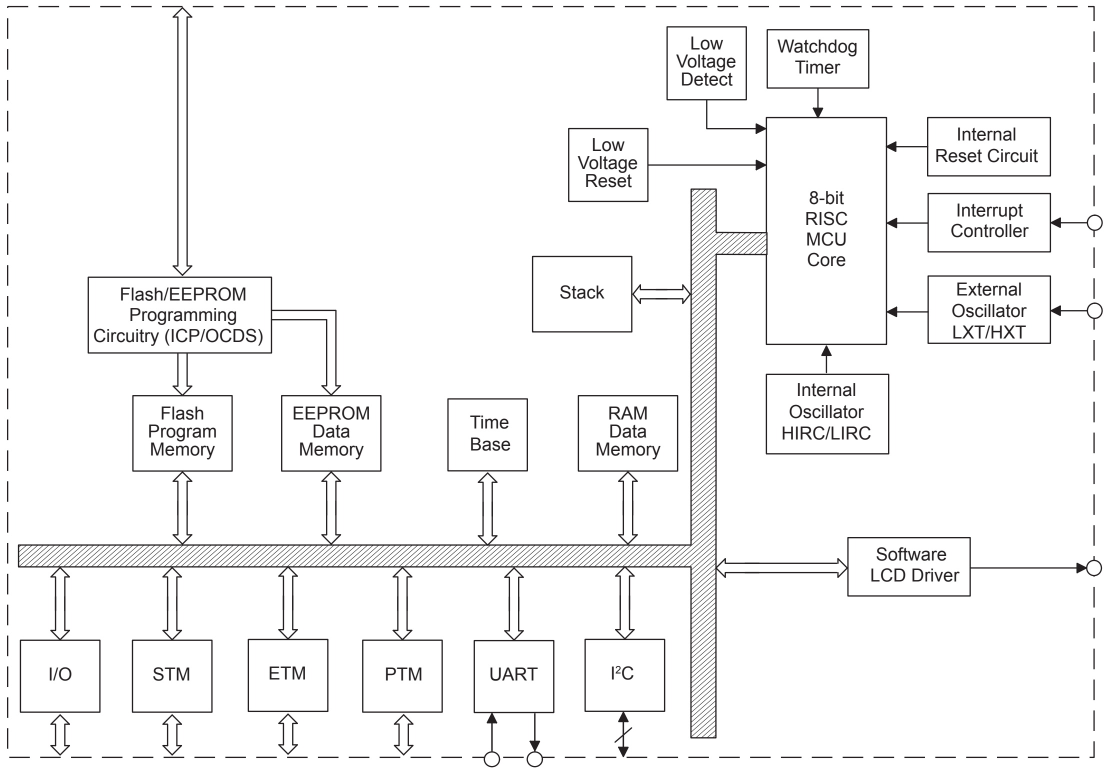
The device is a Flash Memory LCD type 8-bit high performance RISC architecture microcontroller, designed for applications which require an LCD or LED interface. Offering users the convenience of Flash Memory multi-programming features, the device also includes a wide range of functions and features. Other memory includes an area of RAM Data Memory together with an area of EEPROM Data Memory.

Analog features include multiple and extremely flexible Timer Modules provide timing, pulse generation and PWM generation functions. Protective features such as an internal Watchdog Timer, Low Voltage Reset and Low Voltage Detector coupled with excellent noise immunity and ESD protection ensure that reliable operation is maintained in hostile electrical environments.

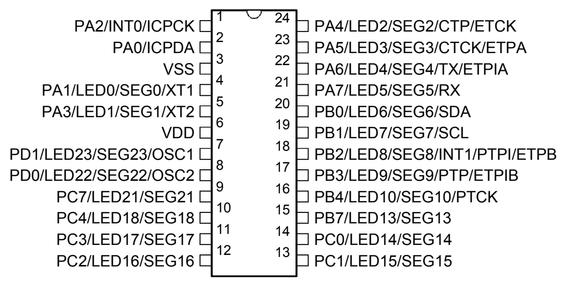
A full choice of internal and external oscillator functions are provided including fully integrated low and high speed system oscillators which requires no external components for their implementation. The ability to operate and switch dynamically between a range of operating modes using different clock sources gives users the ability to optimise microcontroller operation and minimise power consumption.

The inclusion of flexible I/O programming features, Time-Base functions along with many other features ensure that the devices will find excellent use in applications such as consumer products, handheld instruments, household appliances, electronically controlled tools and motor driving in addition to many others.

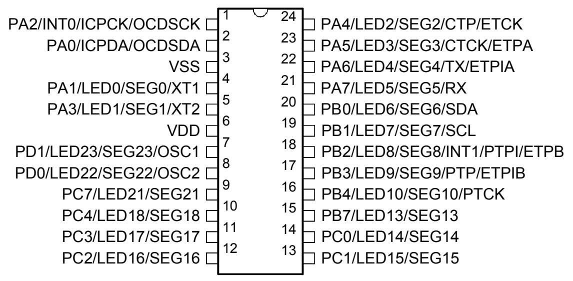
Block Diagram



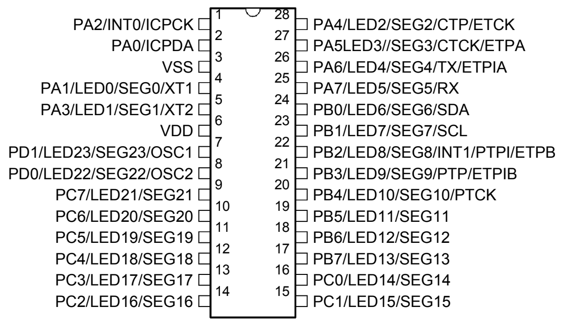
Pin Assignment



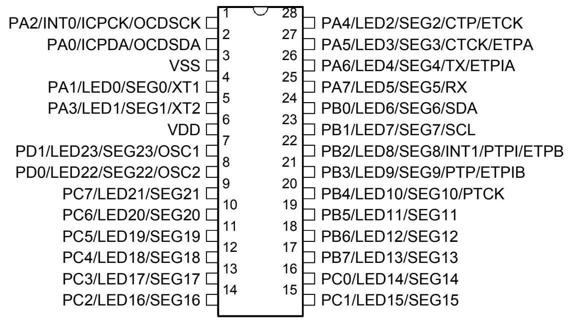
HT69F240
24 SOP-A



HT69V240
24 SOP-A



HT69F240
28 SOP-A



HT69V240
28 SOP-A

Note: 1. If the pin-shared pin functions have multiple outputs simultaneously, the output function is determined by the corresponding software control bits except the functions determined by the configuration options.

2. The HT69V240 device is the EV chip of the HT69F240 device. It supports the "On-Chip Debug" function for debugging during development using the OCDSDA and OCDSCK pins connected to the Holtek HT-IDE development tools.

Pin Description

Pad Name	Function	OPT	I/T	O/T	Description
PA0/ICPDA/ OCDSDA	PA0	PAWU PAPU PAFS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	ICPDA	—	ST	CMOS	ICP Data/Address pin
	OCDSDA	—	ST	CMOS	OCDS Data/Address pin, for EV chip only.
PA1/LED0/ SEG0/XT1	PA1	PAWU PAPU PAFS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED0	PAFS0	—	CMOS	LED output
	SEG0	PAFS0	—	AO	LCD segment output
	XT1	CO	LXT	—	LXT oscillator pin
PA2/INT0/ ICPCK/ OCDSCK	PA2	PAWU PAPU PAFS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	INT0	INTEG INTC0	ST	—	External Interrupt 0
	ICPCK	—	ST	CMOS	ICP Clock pin
	OCDSCK	—	ST	—	OCDS Clock pin, for EV chip only.
PA3/LED1/ SEG1/XT2	PA3	PAWU PAPU PAFS0	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED1	PAFS0	—	CMOS	LED output
	SEG1	PAFS0	—	AO	LCD segment output
	XT2	CO	—	LXT	LXT oscillator pin
PA4/LED2/ SEG2/CTP/ ETCK	PA4	PAPU PAWU	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED2	PAFS1	—	CMOS	LED output
	SEG2	PAFS1	—	AO	LCD segment output
	CTP	PAFS1	—	CMOS	CTM output pin
	ETCK	PAFS1	ST	—	ETM input pin
PA5/LED3/ SEG3/CTCK/ ETPA	PA5	PAWU PAPU PAFS1	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED3	PAFS1	—	CMOS	LED output
	SEG3	PAFS1	—	AO	LCD segment output
	CTCK	PAFS1	ST	—	CTM input pin
	ETPA	PAFS1	—	CMOS	ETM output pin
PA6/LED4/ SEG4/TX/ ETPIA	PA6	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED4	PAFS1	—	CMOS	LED output
	SEG4	PAFS1	—	AO	LCD segment output
	TX	PAFS1	—	CMOS	UART transmit pin
	ETPIA	ETMC1	ST	—	ETM input pin

Pad Name	Function	OPT	I/T	O/T	Description
PA7/LED5/ SEG5/RX	PA7	PAWU PAPU PAFS	ST	CMOS	General purpose I/O. Register enabled pull-up and wake-up.
	LED5	PAFS1	—	CMOS	LED output
	SEG5	PAFS1	—	AO	LCD segment output
	RX	PAFS1	ST	—	UART receive pin
PB0/LED6/ SEG6/SDA	PB0	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED6	PBFS0	—	CMOS	LED output
	SEG6	PBFS0	—	AO	LCD segment output
	SDA	PBFS0	ST	NMOS	I^{2}C data/address line
PB1/LED7/ SEG7/SCL	PB1	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED7	PBFS0	—	CMOS	LED output
	SEG7	PBFS0	—	AO	LCD segment output
	SCL	PBFS0	ST	NMOS	I^{2}C clock line
PB2/LED8/ SEG8/INT1/ PTPI/ETPB	PB2	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED8	PBFS0	—	CMOS	LED output
	SEG8	PBFS0	—	AO	LCD segment output
	INT1	INTEG INTC0	ST	—	External Interrupt 1
	PTPI	PTMC1	ST	—	PTM input pin
	ETPB	PBFS0	—	CMOS	ETM output pin
PB3/LED9/ SEG9/PTP/ ETPIB	PB3	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED9	PBFS0	—	CMOS	LED output
	SEG9	PBFS0	—	AO	LCD segment output
	PTP	PBFS0	—	CMOS	PTM output pin
	ETPIB	ETMC1	ST	—	ETM input pin
PB4/LED10/ SEG10/PTCK	PB4	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED10	PBFS1	—	CMOS	LED output
	SEG10	PBFS1	—	AO	LCD segment output
	PTCK	PBFS1	ST	—	PTM input pin
PB5/LED11/ SEG11	PB5	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED11	PBFS1	—	CMOS	LED output
	SEG11	PBFS1	—	AO	LCD segment output
PB6/LED12/ SEG12	PB6	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED12	PBFS1	—	CMOS	LED output
	SEG12	PBFS1	—	AO	LCD segment output
PB7/LED13/ SEG13	PB7	PBPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED13	PBFS1	—	CMOS	LED output
	SEG13	PBFS1	—	AO	LCD segment output
PC0/LED14/ SEG14	PC0	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED14	PCFS0	—	CMOS	LED output
	SEG14	PCFS0	—	AO	LCD segment output
PC1/LED15/ SEG15	PC1	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED15	PCFS0	—	CMOS	LED output
	SEG15	PCFS0	—	AO	LCD segment output
PC2/LED16/ SEG16	PC2	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED16	PCFS0	—	CMOS	LED output
	SEG16	PCFS0	—	AO	LCD segment output

Pad Name	Function	OPT	I/T	O/T	Description
PC3/LED17/SEG17	PC3	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED17	PCFS0	—	CMOS	LED output
	SEG17	PCFS0	—	AO	LCD segment output
PC4/LED18/SEG18	PC4	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED18	PCFS1	—	CMOS	LED output
	SEG18	PCFS1	—	AO	LCD segment output
PC5/LED19/SEG19	PC5	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED19	PCFS1	—	CMOS	LED output
	SEG19	PCFS1	—	AO	LCD segment output
PC6/LED20/SEG20	PC6	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED20	PCFS1	—	CMOS	LED output
	SEG20	PCFS1	—	AO	LCD segment output
PC7/LED21/SEG21	PC7	PCPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED21	PCFS1	—	CMOS	LED output
	SEG21	PCFS1	—	AO	LCD segment output
PD0/LED22/SEG22/OSC2	PD0	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED22	PDFS0	—	CMOS	LED output
	SEG22	PDFS0	—	AO	LCD segment output
	OSC2	CO	—	HXT	HXT oscillator pin
PD1/LED23/SEG23/OSC1	PD1	PDPU	ST	CMOS	General purpose I/O. Register enabled pull-up
	LED23	PDFS0	—	CMOS	LED output
	SEG23	PDFS0	—	AO	LCD segment output
	OSC1	CO	HXT	—	HXT oscillator pin
VDD	VDD	—	PWR	—	Positive Power supply.
VSS	VSS	—	PWR	—	Negative Power supply. Ground.

Legend: I/T: Input type
O/T: Output type
OPT: Optional by configuration option (CO) or register option
PWR: Power
CO: Configuration option
ST: Schmitt Trigger input
AO: Analog output
CMOS: CMOS output
NMOS: NMOS output
HXT: High frequency crystal oscillator
LXT: High frequency crystal oscillator

Absolute Maximum Ratings

Supply Voltage	V_{SS}-0.3V to V_{SS}+6.0V
Input Voltage	V_{SS}-0.3V to V_{DD}+0.3V
I_{OL} Total	80mA
Total Power Dissipation	500mW
Storage Temperature	-50°C to 125°C
Operating Temperature	-40°C to 85°C
I_{OH} Total	-80mA

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to the device. Functional operation of this device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

D.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	f_{SYS}=4MHz	2.2	—	5.5	V
			f_{SYS}=8MHz	2.2	—	5.5	V
			f_{SYS}=12MHz	2.7	—	5.5	V
			f_{SYS}=16MHz	4.5	—	5.5	V
I_{DD1}	Operating Current (f_{SYS}=f_H, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, f_H=455kHz, WDT enable	—	100	150	μA
		5V		—	280	450	μA
		3V	No load, f_H=1MHz, WDT enable	—	250	400	uA
		5V		—	500	1000	μA
		3V	No load, f_H=4MHz, WDT enable	—	420	630	μA
		5V		—	700	1500	μA
		3V	No load, f_H=8MHz, WDT enable	—	0.8	1.5	mA
		5V		—	1.5	3.0	mA
		3V	No load, f_H=12MHz, WDT enable	—	1.5	2.5	mA
		5V		—	3.0	5.0	mA
		5V	No load, f_H=16MHz, WDT enable	—	4.0	6.0	mA
I_{DD2}	Operating Current (HXT/HIRC, f_{SYS}=f_L, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, f_H=8MHz, f_L=f_H/2, WDT enable	—	500	750	μA
		5V		—	800	1200	μA
		3V	No load, f_H=8MHz, f_L=f_H/4, WDT enable	—	420	630	μA
		5V		—	700	1000	μA
		3V	No load, f_H=8MHz, f_L=f_H/8, WDT enable	—	400	600	μA
		5V		—	600	800	μA
		3V	No load, f_H=8MHz, f_L=f_H/16, WDT enable	—	360	540	μA
		5V		—	560	700	μA
		3V	No load, f_H=8MHz, f_L=f_H/32, WDT enable	—	320	480	μA
		5V		—	520	650	μA
		3V	No load, f_H=8MHz, f_L=f_H/64, WDT enable	—	280	420	μA
		5V		—	440	600	μA
I_{DD3}	Operating Current (LXT/LIRC, f_{SYS}=f_L=f_{RTC}, f_{SUB}=f_{RTC})	3V	No load, WDT enable	—	10	20	μA
		5V		—	30	50	μA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
I_{STB1}	Standby Current (SLEEP0) (f_{SYS}=off, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, system HALT, WDT disable	—	0.1	1	μA
		5V		—	0.3	2	μA
I_{STB2}	Standby Current (SLEEP1) (f_{SYS}=off, f_{SUB}=f_{RTC}/f_{LIRC})	3V	No load, system HALT, WDT disable, LXTLP=1	—	2	3	μA
		5V		—	3	5	μA
I_{STB3}	Standby Current (IDLE0) (f_{SYS}=off, f_{SUB}=f_{LIRC})	3V	No load, system HALT, WDT disable	—	3	5	μA
		5V		—	5	10	μA
I_{STB4}	Standby Current (IDLE0) (f_{SYS}=off, f_{SUB}=f_{RTC})	3V	No load, system HALT, WDT disable, LXTLP=1	—	2	4	μA
		5V		—	4	8	μA
I_{STB5}	Standby Current (IDLE1) (HXT/HIRC, f_{SYS}=f_H, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, system HALT, WDT enable, f_{SYS}=4MHz, FSYSON=1	—	150	200	μA
		5V		—	200	400	μA
I_{STB6}	Standby Current (IDLE1) (HXT/HIRC, f_{SYS}=f_H, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, system HALT, WDT enable, f_{SYS}=8MHz, FSYSON=1	—	300	400	μA
		5V		—	400	600	μA
I_{STB7}	Standby Current (IDLE1) (HXT/HIRC, f_{SYS}=f_H, f_{SUB}=f_{RTC} or f_{LIRC})	3V	No load, system HALT, WDT enable, f_{SYS}=12MHz, FSYSON=1	—	450	600	μA
		5V		—	600	900	μA
V_{IL1}	Input Low Voltage	—	—	0V	—	0.3V_{DD}	V
V_{IH1}	Input High Voltage	—		0.7V_{DD}	—	V_{DD}	V
V_{SEGH}	Software LCD Segment Output High	—	No load	0.645 V_{DD}	0.67 V_{DD}	0.695 V_{DD}	V
V_{SEGL}	Software LCD Segment Output Low	—	No load	0.305 V_{DD}	0.33 V_{DD}	0.355 V_{DD}	V
I_{OL}	I/O Port Sink Current	3V	V_{OL}=0.1V_{DD}	6	12	—	mA
		5V	V_{OL}=0.1V_{DD}	16	32	—	mA
I_{OH1}	I/O Port Source Current (PA0 & PA2)	3V	V_{OH}=0.9V_{DD}	-3	-6	—	mA
		5V	V_{OH}=0.9V_{DD}	-8	-16	—	mA
I_{OH2}	I/O Port Source Current (Except PA0/PA2)	3V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=00; n=0, 1; m=0, 2, 4, 6	-1.0	-2.0	—	mA
		5V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=00; n=0, 1; m=0, 2, 4, 6	-2.5	-5.0	—	mA
		3V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=01; n=0, 1; m=0, 2, 4, 6	-1.5	-3.0	—	mA
		5V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=01; n=0, 1; m=0, 2, 4, 6	-4.0	-8.0	—	mA
		3V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=10; n=0, 1; m=0, 2, 4, 6	-2.5	-5.0	—	mA
		5V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=10; n=0, 1; m=0, 2, 4, 6	-6.0	-12.0	—	mA
		3V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=11; n=0, 1; m=0, 2, 4, 6	-5.0	-10.0	—	mA
		5V	V_{OH}=0.9V_{DD}, PXPSn[m+1:m]=11; n=0, 1; m=0, 2, 4, 6	-13	-26	—	mA
I_{SLCD}	Software LCD Driver Operating Current	5V	ISEL[1:0]=00	5	8	11	μA
			ISEL[1:0]=01	10	16	22	μA
			ISEL[1:0]=10	35	50	65	μA
			ISEL[1:0]=11	70	100	130	μA
R_{PH}	Pull-high Resistance of I/O Ports	3V	—	20	60	100	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Condition				
f_{CPU}	Operating Clock	—	2.2V~5.5V	DC	—	4	MHz
			2.2V~5.5V	DC	—	8	MHz
			2.7V~5.5V	DC	—	12	MHz
			4.5V~5.5V	DC	—	16	MHz
f_{SYS}	System Clock (HXT)	2.2V~5.5V	—	0.4	—	8	MHz
		2.7V~5.5V		0.4	—	12	MHz
		4.5V~5.5V		0.4	—	16	MHz
f_{HIRC}	System Clock (HIRC)	3.0V/5.0V	Trim@3V/5V, Ta=25°C	-2%	4	+2%	MHz
		5.0V	Trim@5V, Ta=25°C	-2%	8	+2%	MHz
		5.0V	Trim@5V, Ta=25°C	-2%	12	+2%	MHz
		2.2V~5.5V	Trim@3V/5V, Ta=-40~85°C	-5%	4	+5%	MHz
		3.0V~5.5V	Trim@5V, Ta=-40~85°C	-5%	8	+5%	MHz
		4.5V~5.5V	Trim@5V, Ta=-40~85°C	-5%	12	+5%	MHz
f_{LXT}	System Clock (LXT)	—	—	—	32768	—	Hz
f_{LIRC}	System Clock (LIRC)	5.0V	Ta = 25°C	-10%	32	+10%	kHz
t_{TCK}	TCK Input Pulse Width	—	—	0.3	—	—	μs
t_{INT}	Interrupt Input Pulse Width	—	—	10	—	—	μs
t_{SST}	System Start-up Timer Period (Wake-up from HALT, f_{SYS} off at HALT state, Slow Mode → Normal Mode, Normal Mode → Slow Mode)	—	f_{SYS} =HXT or LXT	1024	—	—	t_{SYS}
		—	f_{SYS} = HIRC	16	—	—	t_{SYS}
		—	f_{SYS} = LIRC	2	—	—	t_{SYS}
	System Start-up Timer Period (Wake-up from HALT, f_{SYS} on at HALT State)	—	—	2	—	—	t_{SYS}
t_{RSTD}	System Reset Delay Time (Power On Reset, LVR Reset, LVR S/W Reset (LVRC), WDT S/W Reset (WDTC))	—	—	25	50	100	ms
	System Reset Delay Time (WDT Normal Reset)	—	—	8.3	16.7	33.3	ms
t_{SRESET}	Software Reset Width to Reset	—	—	20	45	90	μs
t_{EERD}	EEPROM Read Time	—	—	1	2	4	t_{SYS}
t_{EEWR}	EEPROM Write Timet	—	—	1	2	4	ms

Note: 1. t_{SYS}= 1/f_{SYS}; t_{LIRC}= 1/f_{LIRC}

LVD & LVR Electrical Characteristics

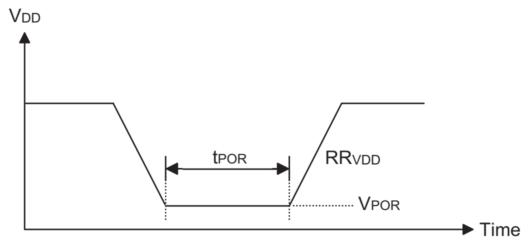
Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{LVR}	Low Voltage Reset Voltage	—	LVR Enable, 2.10V option	-5%	2.1	+5%	V
			LVR Enable, 2.55V option		2.55		
			LVR Enable, 3.15V option		3.15		
			LVR Enable, 3.80V option		3.8		
V_{LVD}	Low Voltage Detector Voltage	—	LVDEN=1, V_{LVD}=2.0V	-5%	2.0	+5%	V
			LVDEN=1, V_{LVD}=2.2V		2.2		V
			LVDEN=1, V_{LVD}=2.4V		2.4		V
			LVDEN=1, V_{LVD}=2.7V		2.7		V
			LVDEN=1, V_{LVD}=3.0V		3.0		V
			LVDEN=1, V_{LVD}=3.3V		3.3		V
			LVDEN=1, V_{LVD}=3.6V		3.6		V
			LVDEN=1, V_{LVD}=4.0V		4.0		V
I_{LVR}	Additional Power Consumption if LVR is used	3.0V	LVR disable → LVR enable	—	10	20	μA
		5.0V		—	15	30	μA
I_{LVD}	Additional Power Consumption if LVD is used	3.0V	LVD disable → LVD enable (LVR disable)	—	10	20	μA
		5.0V		—	15	30	μA
		3.0V	LVD disable → LVD enable (LVR enable)	—	1	2	μA
		5.0V		—	2	4	μA
t_{LVR}	Low Voltage Width to Reset	—	—	120	240	480	μs
t_{LVD}	Low Voltage Width to Interrupt	—	—	20	45	90	μs
t_{LVDS}	LVDO Stable Time	—	For LVR enable, LVD off→on	—	—	15	μs
		—	For LVR disable, LVD off→on	—	—	150	μs

Power-on Reset Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{POR}	V_{DD} Start Voltage to Ensure Power-on Reset	—	—	—	—	100	mV
RR_{VDD}	V_{DD} Raising Rate to Ensure Power-on Reset	—	—	0.035	—	—	V/ms
t_{POR}	Minimum Time for V_{DD} Stays at V_{POR} to Ensure Power-on Reset	—	—	1	—	—	ms



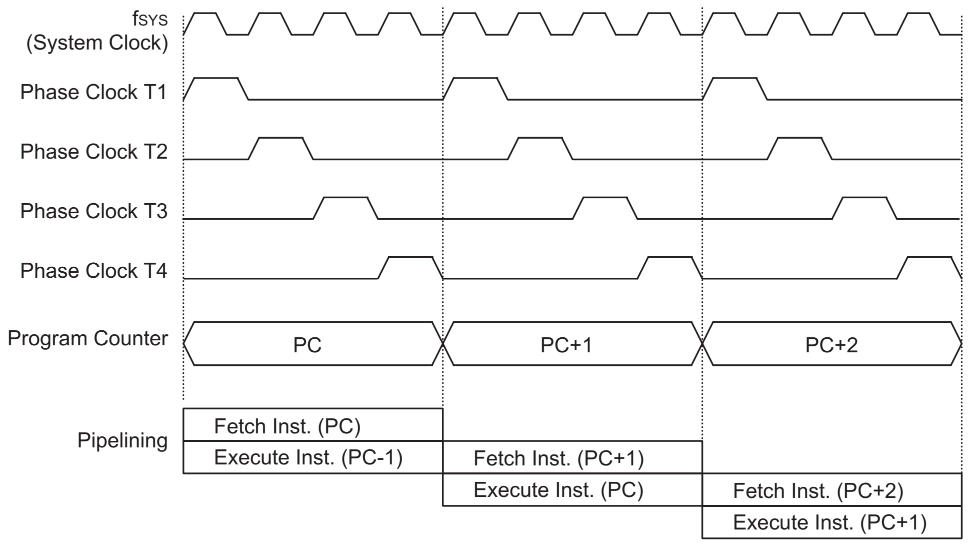
System Architecture

A key factor in the high-performance features of the Holtek range of microcontrollers is attributed to their internal system architecture. The range of devices take advantage of the usual features found within RISC microcontrollers providing increased speed of operation and enhanced performance. The pipelining scheme is implemented in such a way that instruction fetching and instruction execution are overlapped, hence instructions are effectively executed in one cycle, with the exception of branch or call instructions. An 8-bit wide ALU is used in practically all instruction set operations, which carries out arithmetic operations, logic operations, rotation, increment, decrement, branch decisions, etc. The internal data path is simplified by moving data through the Accumulator and the ALU. Certain internal registers are implemented in the Data Memory and can be directly or indirectly addressed. The simple addressing methods of these registers along with additional architectural features ensure that a minimum of external components is required to provide a functional I/O control system with maximum reliability and flexibility. This makes the device suitable for low-cost, high-volume production for controller applications.

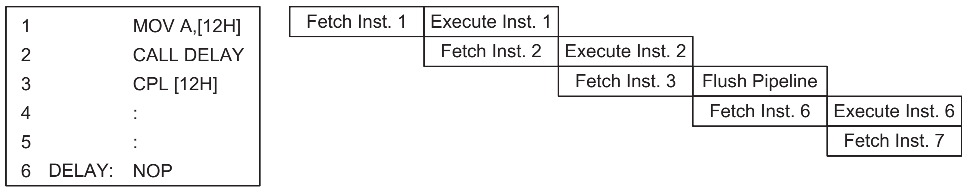
Clocking and Pipelining

The main system clock, derived from either a HXT, LXT, HIRC or LIRC oscillator is subdivided into four internally generated non-overlapping clocks, T1~T4. The Program Counter is incremented at the beginning of the T1 clock during which time a new instruction is fetched. The remaining T2~T4 clocks carry out the decoding and execution functions. In this way, one T1~T4 clock cycle forms one instruction cycle. Although the fetching and execution of instructions takes place in consecutive instruction cycles, the pipelining structure of the microcontroller ensures that instructions are effectively executed in one instruction cycle. The exception to this are instructions where the contents of the Program Counter are changed, such as subroutine calls or jumps, in which case the instruction will take one more instruction cycle to execute.

For instructions involving branches, such as jump or call instructions, two machine cycles are required to complete instruction execution. An extra cycle is required as the program takes one cycle to first obtain the actual jump or call address and then another cycle to actually execute the branch. The requirement for this extra cycle should be taken into account by programmers in timing sensitive applications.



System Clocking and Pipelining



Instruction Fetching

Program Counter

During program execution, the Program Counter is used to keep track of the address of the next instruction to be executed. It is automatically incremented by one each time an instruction is executed except for instructions, such as "JMP" or "CALL" that demand a jump to a non-consecutive Program Memory address. Only the lower 8 bits, known as the Program Counter Low Register, are directly addressable by the application program.

When executing instructions requiring jumps to non-consecutive addresses such as a jump instruction, a subroutine call, interrupt or reset, etc., the microcontroller manages program control by loading the required address into the Program Counter. For conditional skip instructions, once the condition has been met, the next instruction, which has already been fetched during the present instruction execution, is discarded and a dummy cycle takes its place while the correct instruction is obtained.

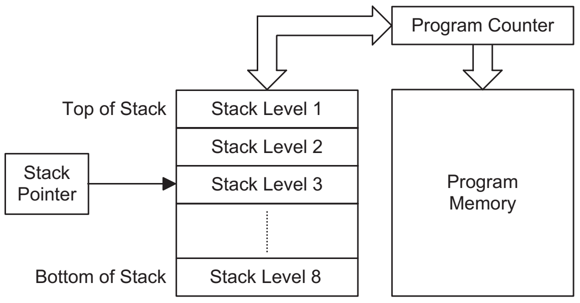
The lower byte of the Program Counter, known as the Program Counter Low register or PCL, is available for program control and is a readable and writeable register. By transferring data directly into this register, a short program jump can be executed directly, however, as only this low byte is available for manipulation, the jumps are limited to the present page of memory, that is 256 locations. When such program jumps are executed it should also be noted that a dummy cycle will be inserted. Manipulating the PCL register may cause program branching, so an extra cycle is needed to pre-fetch.

Stack

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack has multiple levels depending upon the device and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the Stack Pointer, and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, the contents of the Program Counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return instruction, RET or RETI, the Program Counter is restored to its previous value from the stack. After a device reset, the Stack Pointer will point to the top of the stack.

If the stack is full and an enabled interrupt takes place, the interrupt request flag will be recorded but the acknowledge signal will be inhibited. When the Stack Pointer is decremented, by RET or RETI, the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. However, when the stack is full, a CALL subroutine instruction can still be executed which will result in a stack overflow. Precautions should be taken to avoid such cases which might cause unpredictable program branching.

If the stack is overflow, the first Program Counter save in the stack will be lost.



Arithmetic and Logic Unit – ALU

The arithmetic-logic unit or ALU is a critical area of the microcontroller that carries out arithmetic and logic operations of the instruction set. Connected to the main microcontroller data bus, the ALU receives related instruction codes and performs the required arithmetic or logical operations after which the result will be placed in the specified register. As these ALU calculation or operations may result in carry, borrow or other status changes, the status register will be correspondingly updated to reflect these changes. The ALU supports the following functions:

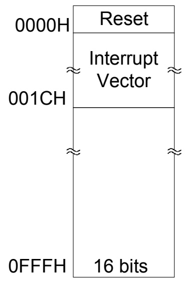
- Arithmetic operations: ADD, ADDM, ADC, ADCM, SUB, SUBM, SBC, SBCM, DAA
- Logic operations: AND, OR, XOR, ANDM, ORM, XORM, CPL, CPLA
- Rotation RRA, RR, RRCA, RRC, RLA, RL, RLCA, RLC
- Increment and Decrement INCA, INC, DECA, DEC
- Branch decision, JMP, SZ, SZA, SNZ, SIZ, SDZ, SIZA, SDZA, CALL, RET, RETI

Flash Program Memory

The Program Memory is the location where the user code or program is stored. For this device the Program Memory is Flash type, which means it can be programmed and re-programmed a large number of times, allowing the user the convenience of code modification on the same device. By using the appropriate programming tools, the Flash device offers users the flexibility to conveniently debug and develop their applications while also offering a means of field programming and updating.

Structure

The Program Memory has a capacity of up to 4k×16 bits. The Program Memory is addressed by the Program Counter and also contains data, table information and interrupt entries information. Table data, which can be setup in any location within the Program Memory, is addressed by separate table pointer registers.



Program Memory Structure

Special Vectors

Within the Program Memory, certain locations are reserved for the reset and interrupts. The location 000H is reserved for use by the device reset for program initialisation. After a device reset is initiated, the program will jump to this location and begin execution.

Look-up Table

Any location within the Program Memory can be defined as a look-up table where programmers can store fixed data. To use the look-up table, the table pointer must first be setup by placing the address of the look up data to be retrieved in the table pointer register, TBLP and TBHP. These registers define the total address of the look-up table.

After setting up the table pointer, the table data can be retrieved from the Program Memory using the "TABRD [m]" or "TABRDL [m]" instructions, respectively. When the instruction is executed, the lower order table byte from the Program Memory will be transferred to the user defined Data Memory register [m] as specified in the instruction. The higher order table data byte from the Program Memory will be transferred to the TBLH special register. Any unused bits in this transferred higher order byte will be read as 0.

The accompanying diagram illustrates the addressing data flow of the look-up table.

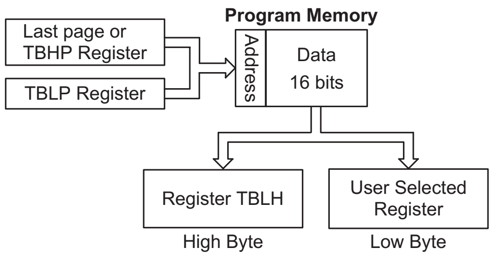


Table Program Example

The accompanying example shows how the table pointer and table data is defined and retrieved from the device. This example uses raw table data located in the last page which is stored there using the ORG statement. The value at this ORG statement is "0F00H" which refers to the start address of the last page within the 4K Program Memory of the device. The table pointer is setup here to have an initial value of "06H". This will ensure that the first data read from the data table will be at the Program Memory address "0F06H" or 6 locations after the start of the last page. Note that the value for the table pointer is referenced to the first address of the present page if the "TABRD [m]" instruction is being used. The high byte of the table data which in this case is equal to zero will be transferred to the TBLH register automatically when the "TABRD [m]" instruction is executed.

Because the TBLH register is a read-only register and cannot be restored, care should be taken to ensure its protection if both the main routine and Interrupt Service Routine use table read instructions. If using the table read instructions, the Interrupt Service Routines may change the value of the TBLH and subsequently cause errors if used again by the main routine. As a rule it is recommended that simultaneous use of the table read instructions should be avoided. However, in situations where simultaneous use cannot be avoided, the interrupts should be disabled prior to the execution of any main routine table-read instructions. Note that all table related instructions require two instruction cycles to complete their operation.

Table Read Program Example

```
tempreg1 db ?            ; temporary register #1
tempreg2 db ?            ; temporary register #2
        :
        :
mov  a,06h               ; initialise table pointer - note that this address
                         ; is referenced to the last page or present page
mov  tblp, a
        :
        :
tabrdl   tempreg1        ; transfers value in table referenced by table pointer
                         ; to tempreg1
                         ; data at program memory address "0F06H" transferred to
                         ; to tempreg1 and TBLH
dec  tblp                ; reduce value of table pointer by one
tabrdl   tempreg2        ; transfers value in table referenced by table pointer
                         ; to tempreg2
                         ; data at program memory address "0F05H" transferred to
                         ; tempreg2 and TBLH
                         ; in this example the data "1AH" is transferred to
                         ; tempreg1 and data "0FH" to register tempreg2
                         ; the value "00H" will be transferred to the high byte
                         ; register TBLH
        :
        :
org 0F00h                   ; sets initial address of last page
dc 00Ah, 00Bh, 00Ch, 00Dh, 00Eh, 00Fh, 01Ah, 01Bh
        :
        :
```

In Circuit Programming – ICP

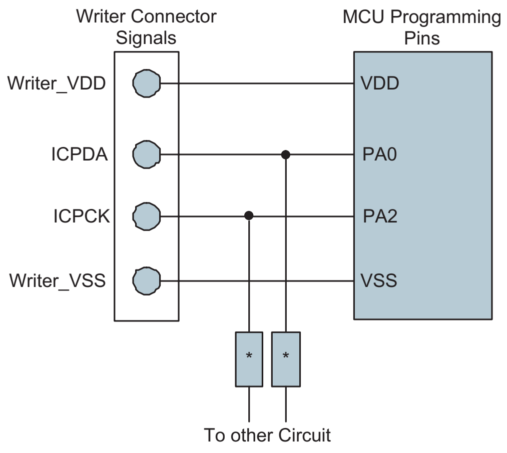
The provision of Flash type Program Memory provides the user with a means of convenient and easy upgrades and modifications to their programs on the same device.

As an additional convenience, Holtek has provided a means of programming the microcontroller in-circuit using a 4-pin interface. This provides manufacturers with the possibility of manufacturing their circuit boards complete with a programmed or un-programmed microcontroller, and then programming or upgrading the program at a later stage. This enables product manufacturers to easily keep their manufactured products supplied with the latest program releases without removal and re-insertion of the device.

Holtek Writer Pins	MCU Programming Pins	Pin Description
ICPDA	PA0	Programming Serial Data/Address
ICPCK	PA2	Programming Clock
VDD	VDD	Power Supply
VSS	VSS	Ground

The Program Memory can be programmed serially in-circuit using this 4-wire interface. Data is downloaded and uploaded serially on a single pin with an additional line for the clock. Two additional lines are required for the power supply and one line for the reset. The technical details regarding the in-circuit programming of the devices are beyond the scope of this document and will be supplied in supplementary literature.

During the programming process, taking control of the ICPDA and ICPCK pins for data and clock programming purposes. The user must there take care to ensure that no other outputs are connected to these two pins.



Note: * may be resistor or capacitor. The resistance of * must be greater than 1k or the capacitance of * must be less than 1nF.

On-Chip Debug Support – OCDS

There is an EV chip device, HT69V240, which is used to emulate the real MCU device, HT69F240. The EV chip device also provides the "On-Chip Debug" function to debug the real MCU device during development process. The real MCU and EV chip devices are almost functional compatible except the "On-Chip Debug" function and package types. Users can use the EV chip device to emulate the real MCU device behaviors by connecting the OCDSDA and OCDSCK pins to the Holtek HT-IDE development tools. The OCDSDA pin is the OCDS Data/Address input/output pin while the OCDSCK pin is the OCDS clock input pin. When users use the EV chip device for debugging, the corresponding pin functions shared with the OCDSDA and OCDSCK pins in the real MCU device will have no effect in the EV chip device. However, the two OCDS pins which are pin-shared with the ICP programming pins are still used as the Flash Memory programming pins for ICP. For more detailed OCDS information, refer to the corresponding document named "Holtek e-Link for 8-bit MCU OCDS User's Guide".

Holtek e-Link Pins	EV Chip Pins	Pin Description
OCDSDA	OCDSDA	On-Chip Debug Support Data/Address input/output
OCDSCK	OCDSCK	On-Chip Debug Support Clock input
VDD	VDD	Power Supply
GND	VSS	Ground

Data Memory

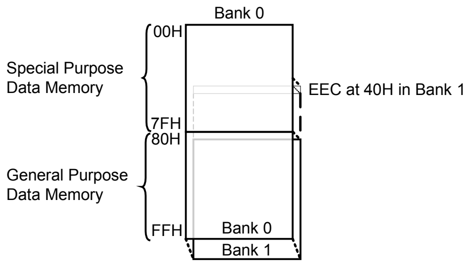
The Data Memory is an 8-bit wide RAM internal memory and is the location where temporary information is stored.

Divided into two sections, the first of these is an area of RAM where special function registers are located. These registers have fixed locations and are necessary for correct operation of the device. Many of these registers can be read from and written to directly under program control, however, some remain protected from user manipulation. The second area of Data Memory is reserved for general purpose use. All locations within this area are read and write accessible under program control. Switching between the different Data Memory banks is achieved by setting the Bank Pointer to the correct value.

Structure

The Data Memory is subdivided into two banks, all of which are implemented in 8-bit wide RAM. The Data Memory located in Bank 0 is subdivided into two sections, the Special Purpose Data Memory and the General Purpose Data Memory. Registers which are located in the Special purpose data memory and common to all microcontrollers, such as ACC, PCL, etc., have the same Data Memory address in all banks except the EEC register located at 40H in Bank 1.

Type	Banks
Special Purpose	0: 00H~7FH 1: 40H(EEC)
General Purpose	0: 80H~FFH 1: 80H~FFH



Data Memory Structure

General Purpose Data Memory

All microcontroller programs require an area of read/write memory where temporary data can be stored and retrieved for use later. It is this area of RAM memory that is known as General Purpose Data Memory. This area of Data Memory is fully accessible by the user programing for both reading and writing operations. By using the "SET [m].i" and "CLR [m].i" instructions individual bits can be set or reset under program control giving the user a large range of flexibility for bit manipulation in the Data Memory.

Special Purpose Data Memory

This area of Data Memory is where registers, necessary for the correct operation of the microcontroller, are stored. Most of the registers are both readable and writeable but some are protected and are readable only, the details of which are located under the relevant Special Function Register section. Note that for locations that are unused, any read instruction to these addresses will return the value "00H".

	Bank 0, 1
00H	IAR0
01H	MP0
02H	IAR1
03H	MP1
04H	BP
05H	ACC
06H	PCL
07H	TBLP
08H	TBLH
09H	TBHP
0AH	STATUS
0BH	SMOD
0CH	LVDC
0DH	INTEG
0EH	WDTC
0FH	TBC
10H	INTC0
11H	INTC1
12H	SMOD1
13H	LVRC
14H	MFI0
15H	MFI1
16H	MFI2
17H	Unused
18H	PAWU
19H	PAPU
1AH	PA
1BH	PAC
1CH	PBPU
1DH	PB
1EH	PBC
1FH	PCPU
20H	PC
21H	PCC
22H	PDPU
23H	PD
24H	PDC
25H	Unused
26H	Unused
27H	Unused
28H	INTC2
29H	RSTC
2AH	Unused
2BH	Unused
2CH	PXPS0
2DH	PXPS1
2EH	Unused
2FH	Unused
30H	Unused
31H	PAFS0
32H	PAFS1
33H	PBFS0
34H	PBFS1
35H	PCFS0
36H	PCFS1
37H	PDFS
38H	Unused
39H	Unused

	Bank 0	Bank 1
3AH	CTMC0	CTMC0
3BH	CTMC1	CTMC1
3CH	CTMDL	CTMDL
3DH	CTMDH	CTMDH
3EH	CTMAL	CTMAL
3FH	CTMAH	CTMAH
40H	▨	EEC
41H	EEA	EEA
42H	EED	EED
43H–47H	Unused	Unused
48H	ETMC0	ETMC0
49H	ETMC1	ETMC1
4AH	ETMC2	ETMC2
4BH	ETMDL	ETMDL
4CH	ETMDH	ETMDH
4DH	ETMAL	ETMAL
4EH	ETMAH	ETMAH
4FH	ETMBL	ETMBL
50H	ETMBH	ETMBH
51H	PTMC0	PTMC0
52H	PTMC1	PTMC1
53H	PTMDL	PTMDL
54H	PTMDH	PTMDH
55H	PTMAL	PTMAL
56H	PTMAH	PTMAH
57H	PTMRPL	PTMRPL
58H	PTMRPH	PTMRPH
59H	IICC0	IICC0
5AH	IICC1	IICC1
5BH	IICD	IICD
5CH	IICA	IICA
5DH	I2CTOC	I2CTOC
5EH	Unused	Unused
5FH	Unused	Unused
60H	SLCDC	SLCDC
61H	USR	USR
62H	UCR1	UCR1
63H	UCR2	UCR2
64H	TXR_RXR	TXR_RXR
65H	BRG	BRG
66H–7FH	Unused	Unused

▨ : Unused, read as 00H

Special Purpose Data Memory

Special Function Register Description

Most of the Special Function Register details will be described in the relevant functional section; however several registers require a separate description in this section.

Indirect Addressing Registers – IAR0, IAR1

The Indirect Addressing Registers, IAR0 and IAR1, although having their locations in normal RAM register space, do not actually physically exist as normal registers. The method of indirect addressing for RAM data manipulation uses these Indirect Addressing Registers and Memory Pointers, in contrast to direct memory addressing, where the actual memory address is specified. Actions on the IAR0 and IAR1 registers will result in no actual read or write operation to these registers but rather to the memory location specified by their corresponding Memory Pointers, MP0 or MP1. Acting as a pair, IAR0 and MP0 can together access data from Bank 0 while the IAR1 and MP1 register pair can access data from any bank. As the Indirect Addressing Registers are not physically implemented, reading the Indirect Addressing Registers indirectly will return a result of "00H" and writing to the registers indirectly will result in no operation.

Memory Pointers – MP0, MP1

Two Memory Pointers, known as MP0 and MP1 are provided. These Memory Pointers are physically implemented in the Data Memory and can be manipulated in the same way as normal registers providing a convenient way with which to address and track data. When any operation to the relevant Indirect Addressing Registers is carried out, the actual address that the microcontroller is directed to, is the address specified by the related Memory Pointer. MP0, together with Indirect Addressing Register, IAR0, are used to access data from Bank 0, while MP1 and IAR1 are used to access data from all banks according to BP register. Direct Addressing can only be used with Bank 0, all other Banks must be addressed indirectly using MP1 and IAR1.

Indirect Addressing Program Example

```
data .section data
adres1   db ?
adres2   db ?
adres3   db ?
adres4   db ?
block    db ?
code .section at 0 code
org 00h

start:
     mov a,04h                ; setup size of block
     mov block,a
     mov a,offset adres1      ; Accumulator loaded with first RAM address
     mov mp0,a                ; setup memory pointer with first RAM address

loop:
     clr IAR0                 ; clear the data at address defined by MP0
     inc mp0                  ; increment memory pointer
     sdz block                ; check if last memory location has been cleared
     jmp loop
     continue:
```

The important point to note here is that in the example shown above, no reference is made to specific RAM addresses.

Bank Pointer – BP

In this device the Data Memory is divided into two banks. Selecting the required Data Memory area is achieved using the Bank Pointer. The Bank Pointer bit 0 is used to select Data Memory Banks 0~1.

The Data Memory is initialised to Bank 0 after a reset, except for a WDT time-out reset in the Power Down Mode, in which case, the Data Memory bank remains unaffected. It should be noted that the Special Function Data Memory is not affected by the bank selection, which means that the Special Function Registers can be accessed from within any bank. Directly addressing the Data Memory will always result in Bank 0 being accessed irrespective of the value of the Bank Pointer. Accessing data from banks other than Bank 0 must be implemented using Indirect addressing.

BP Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	—	DMBP0
R/W	—	—	—	—	—	—	—	R/W
POR	—	—	—	—	—	—	—	0

Bit 7~1	Unimplemented, read as "0"
Bit 0	**DMBP0:** Data memory bank point 0: Bank 0 1: Bank 1

Accumulator – ACC

The Accumulator is central to the operation of any microcontroller and is closely related with operations carried out by the ALU. The Accumulator is the place where all intermediate results from the ALU are stored. Without the Accumulator it would be necessary to write the result of each calculation or logical operation such as addition, subtraction, shift, etc., to the Data Memory resulting in higher programming and timing overheads. Data transfer operations usually involve the temporary storage function of the Accumulator; for example, when transferring data between one user defined register and another, it is necessary to do this by passing the data through the Accumulator as no direct transfer between two registers is permitted.

Program Counter Low Register – PCL

To provide additional program control functions, the low byte of the Program Counter is made accessible to programmers by locating it within the Special Purpose area of the Data Memory. By manipulating this register, direct jumps to other program locations are easily implemented. Loading a value directly into this PCL register will cause a jump to the specified Program Memory location, however, as the register is only 8-bit wide, only jumps within the current Program Memory page are permitted. When such operations are used, note that a dummy cycle will be inserted.

Look-up Table Registers – TBLP, TBHP, TBLH

These three special function registers are used to control operation of the look-up table which is stored in the Program Memory. TBLP and TBHP are the table pointer and indicates the location where the table data is located. Their value must be setup before any table read commands are executed. Their value can be changed, for example using the "INC" or "DEC" instructions, allowing for easy table data pointing and reading. TBLH is the location where the high order byte of the table data is stored after a table read data instruction has been executed. Note that the lower order table data byte is transferred to a user defined location.

Status Register – STATUS

This 8-bit register contains the zero flag (Z), carry flag (C), auxiliary carry flag (AC), overflow flag (OV), power down flag (PDF), and watchdog time-out flag (TO). These arithmetic/logical operation and system management flags are used to record the status and operation of the microcontroller.

With the exception of the TO and PDF flags, bits in the status register can be altered by instructions like most other registers. Any data written into the status register will not change the TO or PDF flag. In addition, operations related to the status register may give different results due to the different instruction operations. The TO flag can be affected only by a system power-up, a WDT time-out or by executing the "CLR WDT" or "HALT" instruction. The PDF flag is affected only by executing the "HALT" or "CLR WDT" instruction or during a system power-up.

The Z, OV, AC and C flags generally reflect the status of the latest operations.

- C is set if an operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
- AC is set if an operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
- Z is set if the result of an arithmetic or logical operation is zero; otherwise Z is cleared.
- OV is set if an operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
- PDF is cleared by a system power-up or executing the "CLR WDT" instruction. PDF is set by executing the "HALT" instruction.
- TO is cleared by a system power-up or executing the "CLR WDT" or "HALT" instruction. TO is set by a WDT time-out.

In addition, on entering an interrupt sequence or executing a subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status registers are important and if the subroutine can corrupt the status register, precautions must be taken to correctly save it.

STATUS Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	TO	PDF	OV	Z	AC	C
R/W	—	—	R	R	R/W	R/W	R/W	R/W
POR	—	—	0	0	x	x	x	x

"x" unknown

Bit 7, 6 Unimplemented, read as "0"

Bit 5 **TO:** Watchdog Time-Out flag
0: After power up or executing the "CLR WDT" or "HALT" instruction
1: A watchdog time-out occurred

Bit 4 **PDF:** Power down flag
0: After power up or executing the "CLR WDT" instruction
1: By executing the "HALT" instruction

Bit 3 **OV:** Overflow flag
0: No overflow
1: An operation results in a carry into the highest-order bit but not a carry out of the highest-order bit or vice versa

Bit 2 **Z:** Zero flag
0: The result of an arithmetic or logical operation is not zero
1: The result of an arithmetic or logical operation is zero

Bit 1 **AC:** Auxiliary flag
0: No auxiliary carry
1: An operation results in a carry out of the low nibbles in addition, or no borrow from the high nibble into the low nibble in subtraction

Bit 0 **C:** Carry flag
0: No carry-out
1: An operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation

C is also affected by a rotate through carry instruction.

EEPROM Data Memory

The EEPROM Data Memory capacity is up to 64×8 bits for this device. Unlike the Program Memory and RAM Data Memory, the EEPROM Data Memory is not directly mapped into memory space and is therefore not directly addressable in the same way as the other types of memory. Read andWrite operations to the EEPROM are carried out in single byte operations using an address and data register in Bank 0 and a single control register in Bank 1.

EEPROM Registers

Three registers control the overall operation of the internal EEPROM Data Memory. These are the address register, EEA, the data register, EED and a single control register, EEC. As both the EEA and EED registers are located in Bank 0, they can be directly accessed in the same was as any other Special Function Register. The EEC register however, being located in Bank1, cannot be addressed directly and can only be read from or written to indirectly using the MP1 Memory Pointer and Indirect Addressing Register, IAR1. Because the EEC control register is located at address 40H in Bank 1, the MP1 Memory Pointer must first be set to the value 40H and the Bank Pointer register, BP, set to the value, 01H, before any operations on the EEC register are executed.

EEPROM Register List

Register Name	Bit							
	7	6	5	4	3	2	1	0
EEA	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
EED	EED7	EED6	EED5	EED4	EED3	EED2	EED1	EED0
EEC	—	—	—	—	WREN	WR	RDEN	RD

EEA Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	EEA5	EEA4	EEA3	EEA2	EEA1	EEA0
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	x	x	x	x	x	x

"x": unknown

Bit 7~6 Unimplemented, read as "0"

Bit 5~0 **EEA5~EEA0:** Data EEPROM address bit 5~bit 0

EED Register

Bit	7	6	5	4	3	2	1	0
Name	EED5	EED4	EED5	EED4	EED3	EED2	EED1	EED0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **EED7~EED0:** Data EEPROM data bit 7~bit 0

EEC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	WREN	WR	RDEN	RD
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4	Unimplemented, read as "0"
Bit 3	**WREN:** Data EEPROM write operation enable 0: Disable 1: Enable This is the Data EEPROM Write Operation Enable bit which must be set high before Datat EEPROM write operations are carried out. Clearing this bit to zero will inhibit Data EEPROM write operations.
Bit 2	**WR:** Data EEPROM write control 0: Write cycle has finished 1: Activate a write cycle This is the Data EEPROM Write Control bit and when set high by the application program will activate a write cycle. This bit will be automatically reset to zero by the hardware after the write cycle has finished. Setting this bit high will have no effect if the WREN bit has not first been set high.
Bit 1	**RDEN:** Data EEPROM read operation enable 0: Disable 1: Enable This is the Data EEPROM Read Operation Enable bit which must be set high before Datat EEPROM read operations are carried out. Clearing this bit to zero will inhibit Data EEPROM read operations.
Bit 0	**RD:** Data EEPROM read control 0: Read cycle has finished 1: Activate a read cycle This is the Data EEPROM Read Control bit and when set high by the application program will activate a read cycle. This bit will be automatically reset to zero by the hardware after the read cycle has finished. Setting this bit high will have no effect if the RDEN bit has not first been set high.

Note: The WREN, WR, RDEN and RD bits can not be set to "1" at the same time in one instruction.
The WR and RD bits can not be set to "1" at the same time.

Reading Data from the EEPROM

To read data from the EEPROM, the read enable bit, RDEN, in the EEC register must first be set high to enable the read function. The EEPROM address of the data to be read must then be placed in the EEAregister. If the RD bit in the EEC register is now set high, a read cycle will be initiated. Setting the RD bit high will not initiate a read operation if the RDEN bit has not been set. When the read cycle terminates, the RD bit will be automatically cleared to zero, after which the data can be read from the EED register. The data will remain in the EED register until another read or write operation is executed. The application program can poll the RD bit to determine when the data is valid for reading.

Writing Data to the EEPROM

The EEPROM address of the data to be written must first be placed in the EEA register and the data placed in the EED register. To write data to the EEPROM, the write enable bit, WREN, in the EEC register must first be set high to enable the write function. After this, the WR bit in the EEC register must be immediately set high to initiate a write cycle. These two instructions must be executed consecutively. The global interrupt bit EMI should also first be cleared before implementing any write operations, and then set again after the write cycle has started. Note that setting the WR bit high will not initiate a write cycle if the WREN bit has not been set. As the EEPROM write cycle is controlled using an internal timer whose operation is asynchronous to microcontroller system clock, a certain time will elapse before the data will have been written into the EEPROM. Detecting when the write cycle has finished can be implemented either by polling the WR bit in the EEC register or by using the EEPROM interrupt. When the write cycle terminates, the WR bit will be automatically cleared to zero by the microcontroller, informing the user that the data has been written to the EEPROM. The application program can therefore poll the WR bit to determine when the write cycle has ended.

Write Protection

Protection against inadvertent write operation is provided in several ways. After the device is powered-on the Write Enable bit in the control register will be cleared preventing any write operations. Also at power-on the Bank Pointer, BP, will be reset to zero, which means that Data Memory Bank 0 will be selected. As the EEPROM control register is located in Bank 1, this adds a further measure of protection against spurious write operations. During normal program operation, ensuring that the Write Enable bit in the control register is cleared will safeguard against incorrect write operations.

EEPROM Interrupt

The EEPROM write interrupt is generated when an EEPROM write cycle has ended. The EEPROM interrupt must first be enabled by setting the DEE bit in the relevant interrupt register. However as the EEPROM is contained within a Multi-function Interrupt, the associated multi-function interrupt enable bit must also be set. When an EEPROM write cycle ends, the DEF request flag and its associated multi-function interrupt request flag will both be set. If the global, EEPROM and Multi-function interrupts are enabled and the stack is not full, a jump to the associated Multi-function Interrupt vector will take place. When the interrupt is serviced only the Multi-function interrupt flag will be automatically reset, the EEPROM interrupt flag must be manually reset by the application program. More details can be obtained in the Interrupt section.

Programming Considerations

Care must be taken that data is not inadvertently written to the EEPROM. Protection can be enhanced by ensuring that the Write Enable bit is normally cleared to zero when not writing. Also the Bank Pointer could be normally cleared to zero as this would inhibit access to Bank 1 where the EEPROM control register exist. Although certainly not necessary, consideration might be given in the application program to the checking of the validity of new write data by a simple read back process.

When writing data the WR bit must be set high immediately after the WREN bit has been set high, to ensure the write cycle executes correctly. The global interrupt bit EMI should also be cleared before a write cycle is executed and then re-enabled after the write cycle starts.

Programming Examples

Reading Data from the EEPROM – Polling Mothod

```
MOV  A, EEPROM_ADRES         ; user defined address
MOV  EEA, A
MOV  A, 040H                 ; setup memory pointer MP1
MOV  MP1, A                  ; MP1 points to EEC register
MOV  A, 01H                  ; setup Bank Pointer
MOV  BP, A
SET  IAR1.1                  ; set RDEN bit, enable read operations
SET  IAR1.0                  ; start Read Cycle - set RD bit
BACK:
SZ   IAR1.0                  ; check for read cycle end
JMP  BACK
CLR  IAR1                    ; disable EEPROM read/write
CLR  BP
MOV  A, EED                  ; move read data to register
MOV  READ_DATA, A
```

Writing Data to the EEPROM – Polling Mothod

```
CLR  EMI
MOV  A, EEPROM_ADRES         ; user defined address
MOV  EEA, A
MOV  A, EEPROM_DATA          ; user defined data
MOV  EED, A
MOV  A, 040H                 ; setup memory pointer MP1
MOV  MP1, A                  ; MP1 points to EEC register
MOV  A, 01H                  ; setup Bank Pointer
MOV  BP, A
SET  IAR1.3                  ; set WREN bit, enable write operations
SET  IAR1.2                  ; Start Write Cycle - set WR bit - executed immediately
                             ; after set WREN bit
SET  EMI
BACK:
SZ   IAR1.2                  ; check for write cycle end
JMP  BACK
CLR  IAR1                    ; disable EEPROM read/write
CLR  BP
```

Oscillator

Various oscillator options offer the user a wide range of functions according to their various application requirements. The flexible features of the oscillator functions ensure that the best optimisation can be achieved in terms of speed and power saving. Oscillator selections and operation are selected through a combination of configuration options and registers.

Oscillator Overview

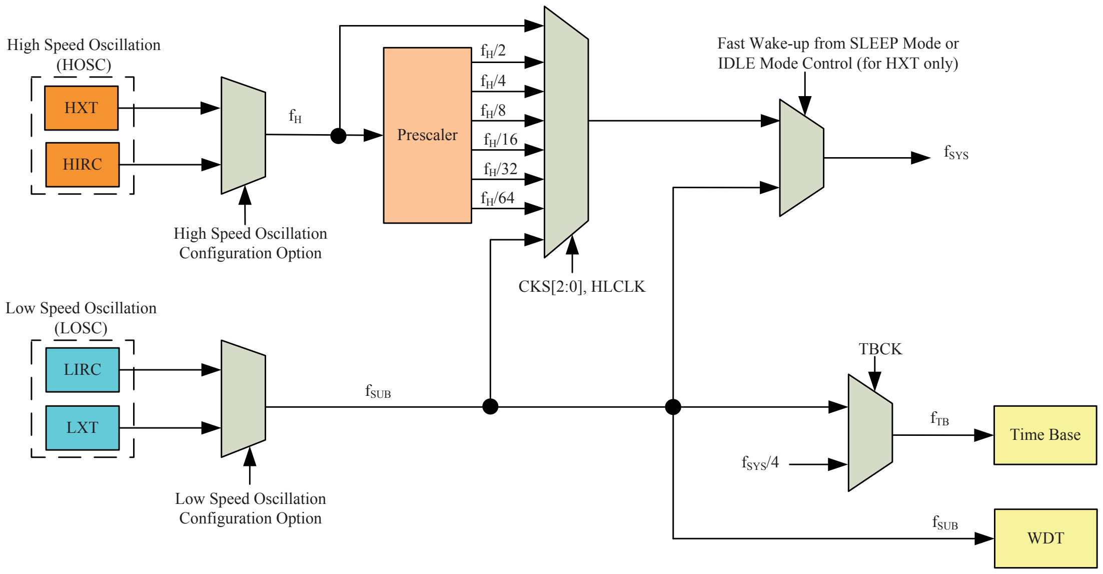
In addition to being the source of the main system clock the oscillators also provide clock sources for the Watchdog Timer and Time Base Interrupts. External oscillators requiring some external components as well as fully integrated internal oscillators, requiring no external components, are provided to form a wide range of both fast and slow system oscillators. All oscillator options are selected through the configuration options. The higher frequency oscillators provide higher performance but carry with it the disadvantage of higher power requirements, while the opposite is of course true for the lower frequency oscillators. With the capability of dynamically switching between fast and slow system clock, the device has the flexibility to optimize the performance/power ratio, a feature especially important in power sensitive portable applications.

Type	Name	Freq.	Pins
External Crystal	HXT	400kHz~16MHz	OSC1/OSC2
Internal High Speed RC	HIRC	4, 8 or 12MHz	—
External Low Speed Crystal	LXT	32.768kHz	XT1/XT2
Internal Low Speed RC	LIRC	32kHz	—

Oscillator Types

System Clock Configurations

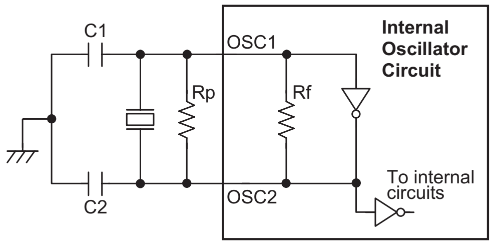
There are four methods of generating the system clock, two high speed oscillators and two low speed oscillators. The high speed oscillators are is the external crystal/ceramic oscillator and the internal 4MHz, 8MHz or 12MHz RC oscillator. The two low speed oscillators are the internal 32kHz RC oscillator and the external 32.768kHz crystal oscillator. Selecting whether the low or high speed oscillator is used as the system oscillator is implemented using the HLCLK bit and CKS2~CKS0 bits in the SMOD register and as the system clock can be dynamically selected.


High Speed Oscillation (HOSC)
HXT
HIRC
f_H
High Speed Oscillation Configuration Option
Prescaler
$f_H/2$
$f_H/4$
$f_H/8$
$f_H/16$
$f_H/32$
$f_H/64$
CKS[2:0], HLCLK
Fast Wake-up from SLEEP Mode or IDLE Mode Control (for HXT only)
f_{SYS}
Low Speed Oscillation (LOSC)
LIRC
LXT
f_{SUB}
Low Speed Oscillation Configuration Option
TBCK
f_{TB}
$f_{SYS}/4$
Time Base
f_{SUB}
WDT

External Crystal/Ceramic Oscillator – HXT

The External Crystal/Ceramic System Oscillator is one of the high frequency oscillator choices, which is selected via configuration option. For most crystal oscillator configurations, the simple connection of a crystal across OSC1 and OSC2 will create the necessary phase shift and feedback for oscillation, without requiring external capacitors. However, for some crystal types and frequencies, to ensure oscillation, it may be necessary to add two small value capacitors, C1 and C2. Using a ceramic resonator will usually require two small value capacitors, C1 and C2, to be connected as shown for oscillation to occur. The values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification. An additional configuration option must be setup to configure the device according to whether the oscillator frequency is high, defined as equal to or above 1MHz, or low, which is defined as below 1MHz.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with inter connecting lines are all located as close to the MCU as possible.



Note: 1. Rp is normally not required. C1 and C2 are required.
2. Although not shown OSC1/OSC2 pins have a parasitic capacitance of around 7pF.

Crystal/Resonator Oscillator – HXT

Crystal Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
12MHz	0pF	0pF
8MHz	0pF	0pF
4MHz	0pF	0pF
1MHz	100pF	100pF
455kHz (see Note2)	100pF	100pF
Note: 1. C1 and C2 values are for guidance only. 2. XTAL mode configuration option: 455kHz.		

Crystal Recommended Capacitor Values

Internal RC Oscillator – HIRC

The internal RC oscillator is a fully integrated system oscillator requiring no external components. The internal RC oscillator has three fixed frequencies of either 4MHz, 8MHz or 12MHz. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of either 3V or 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 4MHz, 8MHz or 12MHz will have a tolerance within 2%. Note that if this internal system clock option is selected, as it requires no external pins for its operation, I/O pins PD0 and PD1 are free for use as normal I/O pins.

External 32.768kHz Crystal Oscillator – LXT

The External 32.768kHz Crystal System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. This clock source has a fixed frequency of 32.768kHz and requires a 32.768kHz crystal to be connected between pins XT1 and XT2. The external resistor and capacitor components connected to the 32.768kHz crystal are necessary to provide oscillation. For applications where precise frequencies are essential, these components may be required to provide frequency compensation due to different crystal manufacturing tolerances. During power-up there is a time delay associated with the LXT oscillator waiting for it to start-up.

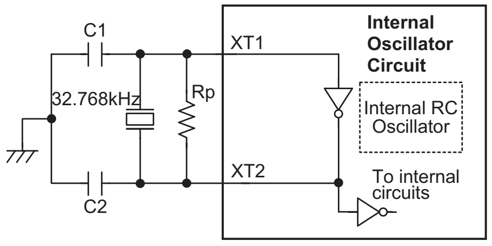
When the microcontroller enters the SLEEP or IDLE Mode, the system clock is switched off to stop microcontroller activity and to conserve power. However, in many microcontroller applications it may be necessary to keep the internal timers operational even when the microcontroller is in the SLEEP or IDLE Mode. To do this, another clock, independent of the system clock, must be provided.

However, for some crystals, to ensure oscillation and accurate frequency generation, it is necessary to add two small value external capacitors, C1 and C2. The exact values of C1 and C2 should be selected in consultation with the crystal or resonator manufacturer's specification. The external parallel feedback resistor, Rp, is required.

Some configuration options determine if the XT1/XT2 pins are used for the LXT oscillator or as I/O pins.

- If the LXT oscillator is not used for any clock source, the XT1/XT2 pins can be used as normal I/O pins.
- If the LXT oscillator is used for any clock source, the 32.768kHz crystal should be connected to the XT1/XT2 pins.

For oscillator stability and to minimise the effects of noise and crosstalk, it is important to ensure that the crystal and any associated resistors and capacitors along with inter connecting lines are all located as close to the MCU as possible.



Note: 1. Rp, C1 and C2 are required.
2. Although not shown pins have a parasitic capacitance of around 7pF.

External LXT Oscillator

LXT Oscillator C1 and C2 Values		
Crystal Frequency	**C1**	**C2**
32.768kHz	10pF	10pF
Note:1. C1 and C2 values are for guidance only. 2. R_P=5M~10MΩ is recommended.		

32.768kHz Crystal Recommended Capacitor Values

LXT Oscillator Low Power Function

The LXT oscillator can function in one of two modes, the Quick Start Mode and the Low Power Mode. The mode selection is executed using the LXTLP bit in the TBC register.

LXTLP Bit	LXT Mode
0	Quick Start
1	Low-power

After power on the LXTLP bit will be automatically cleared to zero ensuring that the LXT oscillator is in the Quick Start operating mode. In the Quick Start Mode the LXT oscillator will power up and stabilise quickly. However, after the LXT oscillator has fully powered up it can be placed into the Low-power mode by setting the LXTLP bit high. The oscillator will continue to run but with reduced current consumption, as the higher current consumption is only required during the LXT oscillator start-up. In power sensitive applications, such as battery applications, where power consumption must be kept to a minimum, it is therefore recommended that the application program sets the LXTLP bit high about 2 seconds after power-on.

It should be noted that, no matter what condition the LXTLP bit is set to, the LXT oscillator will always function normally, the only difference is that it will take more time to start up if in the Low-power mode.

Internal 32kHz Oscillator – LIRC

The Internal 32kHz System Oscillator is one of the low frequency oscillator choices, which is selected via configuration option. It is a fully integrated RC oscillator with a typical frequency of 32kHz at 5V, requiring no external components for its implementation. Device trimming during the manufacturing process and the inclusion of internal frequency compensation circuits are used to ensure that the influence of the power supply voltage, temperature and process variations on the oscillation frequency are minimised. As a result, at a power supply of 5V and at a temperature of 25°C degrees, the fixed oscillation frequency of 32kHz will have a tolerance within 10%.

Supplementary Oscillators

The low speed oscillators, in addition to providing a system clock source are also used to provide a clock source to two other device functions. These are the Watchdog Timer and the Time Base Interrupts.

Operating Modes and System Clocks

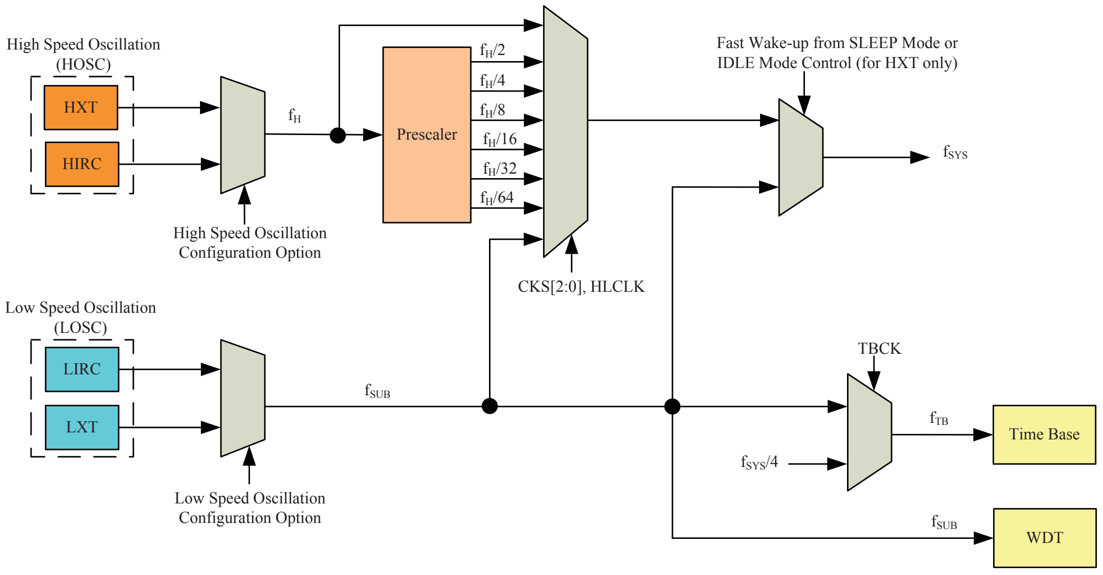
Present day applications require that their microcontrollers have high performance but often still demand that they consume as little power as possible, conflicting requirements that are especially true in battery powered portable applications. The fast clocks required for high performance will by their nature increase current consumption and of course vice-versa, lower speed clocks reduce current consumption. As Holtek has provided these devices with both high and low speed clock sources and the means to switch between them dynamically, the user can optimise the operation of their microcontroller to achieve the best performance/power ratio.

System Clock

The device has many different clock sources for both the CPU and peripheral function operation. By providing the user with a wide range of clock options using configuration options and register programming, a clock system can be configured to obtain maximum application performance.

The main system clock, can come from either a high frequency, f_H, or low frequency, f_{SUB}, source, and is selected using the HLCLK bit and CKS2~CKS0 bits in the SMOD register. The high speed system clock can be sourced from an HXT or HIRC oscillator, selected via a configuration option. The low speed system clock source can be sourced from the clock, f_{SUB}. If f_{SUB} is selected then it can be sourced by either the LXT or LIRC oscillators, selected via a configuration option. The other choice, which is a divided version of the high speed system oscillator has a range of $f_H/2$~$f_H/64$.

The f_{SUB} clock is used to provide a substitute clock for the microcontroller just after a wake-up has occurred to enable faster wake-up times.


High Speed Oscillation (HOSC)
HXT
HIRC
f_H
Prescaler
$f_H/2$
$f_H/4$
$f_H/8$
$f_H/16$
$f_H/32$
$f_H/64$
High Speed Oscillation Configuration Option
CKS[2:0], HLCLK
Fast Wake-up from SLEEP Mode or IDLE Mode Control (for HXT only)
f_{SYS}
Low Speed Oscillation (LOSC)
LIRC
LXT
f_{SUB}
Low Speed Oscillation Configuration Option
TBCK
f_{TB}
Time Base
$f_{SYS}/4$
f_{SUB}
WDT

System Clock Configurations

Note: When the system clock source f_{SYS} is switched to f_{SUB} from f_H, the high speed oscillation will stop to conserve the power. Thus there is no f_H~$f_H/64$ for peripheral circuit to use.

System Operation Modes

There are six different modes of operation for the microcontroller, each one with its own special characteristics and which can be chosen according to the specific performance and power requirements of the application. There are two modes allowing normal operation of the microcontroller, the NORMAL Mode and SLOW Mode. The remaining four modes, the SLEEP0, SLEEP1, IDLE0 and IDLE1 Mode are used when the microcontroller CPU is switched off to conserve power.

Operating Mode	Description		
	CPU	f_{SYS}	f_{SUB}
NORMAL Mode	On	f_H~f_H/64	On
SLOW Mode	On	f_{SUB}	On
IDLE0 Mode	Off	Off	On
IDLE1 Mode	Off	On	On
SLEEP0 Mode	Off	Off	Off
SLEEP1 Mode	Off	Off	On

NORMAL Mode

As the name suggests this is one of the main operating modes where the microcontroller has all of its functions operational and where the system clock is provided by one of the high speed oscillators. This mode operates allowing the microcontroller to operate normally with a clock source will come from one of the high speed oscillators, either the HXT or HIRC oscillators. The high speed oscillator will however first be divided by a ratio ranging from 1 to 64, the actual ratio being selected by the CKS2~CKS0 and HLCLK bits in the SMOD register. Although a high speed oscillator is used, running the microcontroller at a divided clock ratio reduces the operating current.

SLOW Mode

This is also a mode where the microcontroller operates normally although now with a slower speed clock source. The clock source used will be from one of the low speed oscillators, either the LXT or the LIRC. Running the microcontroller in this mode allows it to run with much lower operating currents. In the SLOW Mode, the f_H is off.

SLEEP0 Mode

The SLEEP Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is low. In the SLEEP0 mode the CPU will be stopped, and the f_{SUB} clock will be stopped too, and the Watchdog Timer function is disabled. In this mode, the LVDEN is must set to "0". If the LVDEN is set to "1", it won't enter the SLEEP0 Mode.

SLEEP1 Mode

The SLEEP1 Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is low. In the SLEEP1 mode the CPU will be stopped. However the f_{SUB} will continue to operate if the LVDEN is "1" or the Watchdog Timer function is enabled as its clock source is from the f_{SUB}.

IDLE0 Mode

The IDLE0 Mode is entered when a HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the SMOD1 register is low. In the IDLE0 Mode the system oscillator will be inhibited from driving the CPU but some peripheral functions will remain operational such as the Watchdog Timer and TMs. In the IDLE0 Mode, the system oscillator will be stopped while the Watchdog Timer clock, f_{SUB}, will be on.

IDLE1 Mode

The IDLE1 Mode is entered when an HALT instruction is executed and when the IDLEN bit in the SMOD register is high and the FSYSON bit in the SMOD1 register is high. In the IDLE1 Mode the system oscillator will be inhibited from driving the CPU but may continue to provide a clock source to keep some peripheral functions operational such as Time Base and TMs. In the IDLE1 Mode, the system oscillator will continue to run, and this system oscillator may be high speed or low speed system oscillator. In the IDLE1 Mode the Watchdog Timer clock, f_{SUB}, will also be on.

Control Register

A register pair, SMOD and SMOD1, is used for overall control of the internal clocks within the device.

SMOD Register

Bit	7	6	5	4	3	2	1	0
Name	CKS2	CKS1	CKS0	FSTEN	LTO	HTO	IDLEN	HLCLK
R/W	R/W	R/W	R/W	R/W	R	R	R/W	R/W
POR	0	0	0	0	0	0	1	1

Bit 7~5 **CKS2~CKS0:** The system clock selection when HLCLK is "0"
000: f_{SUB} (f_{LXT} or f_{LIRC})
001: f_{SUB} (f_{LXT} or f_{LIRC})
010: $f_H/64$
011: $f_H/32$
100: $f_H/16$
101: $f_H/8$
110: $f_H/4$
111: $f_H/2$
These three bits are used to select which clock is used as the system clock source. In addition to the system clock source the LIRC, a divided version of the high speed system oscillator can also be chosen as the system clock source.

Bit 4 **FSTEN:** Fast Wake-up Control (only for HXT)
0: Disable
1: Enable
This is the Fast Wake-up Control bit which determines if the f_{SUB} clock source is initially used after the device wakes up. When the bit is high, the f_{SUB} clock source can be used as a temporary system clock to provide a faster wake up time as the f_{SUB} clock is available.

Bit 3 **LTO:** Low speed system oscillator ready flag
0: Not ready
1: Ready
This is the low speed system oscillator ready flag which indicates when the low speed system oscillator is stable after power on reset or a wake-up has occurred. The flag will be low when in the SLEEP0 Mode but after a wake-up has occurred, the flag will change to a high level after 1~2 clock cycles if the LIRC oscillator is used.

Bit 2 **HTO:** High speed system oscillator ready flag
0: Not ready
1: Ready
This is the high speed system oscillator ready flag which indicates when the high speed system oscillator is stable. This flag is cleared to "0" by hardware when the device is powered on and then changes to a high level after the high speed system oscillator is stable. Therefore this flag will always be read as "1" by the application program after device power-on. The flag will be low when in the SLEEP or IDLE0 Mode but after a wake-up has occurred, the flag will change to a high level after 1024 clock cycles if the HXT oscillator is used.

bit 1 **IDLEN:** IDLE Mode control
0: Disable
1: Enable
This is the IDLE Mode Control bit and determines what happens when the HALT instruction is executed. If this bit is high, when a HALT instruction is executed the device will enter the IDLE Mode. In the IDLE1 Mode the CPU will stop running but the system clock will continue to keep the peripheral functions operational, if FSYSON bit is high. If FSYSON bit is low, the CPU and the system clock will all stop in IDLE0 mode. If the bit is low the device will enter the SLEEP Mode when a HALT instruction is executed.

bit 0 **HLCLK:** System clock selection
0: $f_H/2\sim f_H/64$ or f_{SUB}
1: f_H
This bit is used to select if the f_H clock or the $f_H/2\sim f_H/64$ or f_{SUB} clock is used as the system clock. When the bit is high the f_H clock will be selected and if low the $f_H/2\sim f_H/64$ or f_{SUB} clock will be selected. When system clock switches from the f_H clock to the f_{SUB} clock and the f_H clock will be automatically switched off to conserve power.

SMOD1 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	RSTF	LVRF	LRF	WRF
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	x	0	0

"x" unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
0: Disable
1: Enable

Bit 6~4 Unimplemented, read as "0"

Bit 3 **RSTF:** RSTC Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the RSTC register contains any undefined RSTC register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

Bit 2 **LVRF:** LVR function reset flag
0: Not occurred
1: Occurred
This bit is set to 1 when a specific Low Voltage Reset situation occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

bit 0 **WRF:** WDT Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 by the WDT Control register software reset and cleared by the application program. Note that this bit can only be cleared to 0 by the application program.

Fast Wake-up

To minimise power consumption the device can enter the SLEEP or IDLE0 Mode, where the system clock source to the device will be stopped. However when the device is woken up again, it can take a considerable time for the original system oscillator to restart, stabilize and allow normal operation to resume. To ensure the device is up and running as fast as possible a Fast Wake-up function is provided, which allows f_{SUB}, namely the LIRC oscillator, to act as a temporary clock to first drive the system until the original system oscillator has stabilised. As the clock source for the Fast Wake-up function is f_{SUB}, the Fast Wake-up function is only available in the SLEEP1 and IDLE0 modes. When the device is woken up from the SLEEP0 mode, the FastWake-up function has no effect because the f_{SUB} clock is stopped. The FastWake-up enable/disable function is controlled using the FSTEN bit in the SMOD register.

If the HXT oscillator is selected as the NORMAL Mode system clock and if the Fast Wake-up function is enabled, then it will take one to two t_{SUB} clock cycles of the LIRC oscillator for the system to wake-up. The system will then initially run under the f_{SUB} clock source until 1024 HXT clock cycles have elapsed, at which point the HTO flag will switch high and the system will switch over to operating from the HXT oscillator.

If the HIRC or LIRC oscillator is used as the system oscillator, then it will take15~16 clock cycles of the ERC/HIRC oscillator or 1~2 clock cycles of the LIRC osrillator respectively to wake up the system from the SLEEP or IDLE0 Mode. The Fast Wake-up bit, FSTEN will have no effect in these cases.

System Oscillator	FSTEN Bit	Wake-up Time (SLEEP0 Mode)	Wake-up Time (SLEEP1 Mode)	Wake-up Time (IDLE0 Mode)	Wake-up Time (IDLE1 Mode)
HXT	0	1024 HXT cycles	1024 HXT cycles		1~2 HXT cycles
	1	1024 HXT cycles	1~2 f_{SUB} cycles (System runs with f_{SUB} first for 1024 HXT cycles and then switches over to run with the HXT clock)		1~2 HXT cycles
HIRC	x	15~16 HIRC cycles	15~16 HIRC cycles		1~2 HIRC cycles
LIRC	x	1~2 LIRC cycles	1~2 LIRC cycles		1~2 LIRC cycles
LXT	x	1024 LXT cycles	1024 LXT cycles		1~2 LXT cycles

Wake-up Times

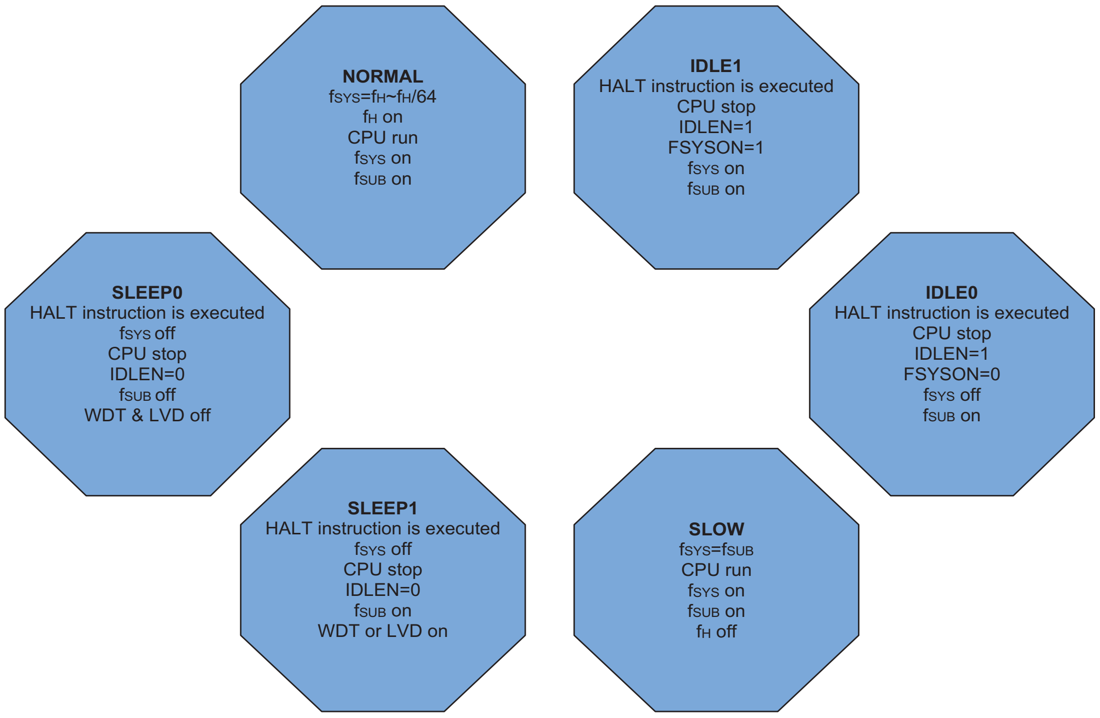
Note that if the Watchdog Timer is disabled, which means that the f_{SUB} clock drived from the LXT or LIRC oscillator is off, then there will be no Fast Wake-up function available when the device wakes-up from the SLEEP0 Mode.

Operating Mode Switching

The device can switch between operating modes dynamically allowing the user to select the best performance/power ratio for the present task in hand. In this way microcontroller operations that do not require high performance can be executed using slower clocks thus requiring less operating current and prolonging battery life in portable applications.

In simple terms, Mode Switching between the NORMAL Mode and SLOW Mode is executed using the HLCLK bit and CKS2~CKS0 bits in the SMOD register while Mode Switching from the NORMAL/SLOW Modes to the SLEEP/IDLE Modes is executed via the HALT instruction. When a HALT instruction is executed, whether the device enters the IDLE Mode or the SLEEP Mode is determined by the condition of the IDLEN bit in the SMOD register and FSYSON in the SMOD1 register.

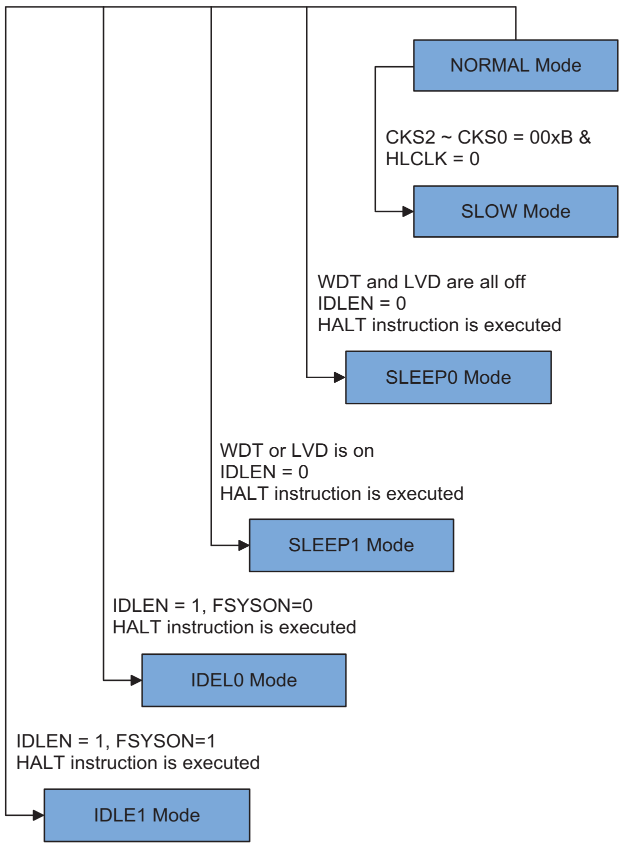
When the HLCLK bit switches to a low level, which implies that clock source is switched from the high speed clock source, f_H, to the clock source, $f_H/2$~$f_H/64$ or f_{SUB}. If the clock is from the f_{SUB}, the high speed clock source will stop running to conserve power. When this happens it must be noted that the $f_H/16$ and $f_H/64$ internal clock sources will also stop running, which may affect the operation of other internal functions such as the TMs. The accompanying flowchart shows what happens when the device moves between the various operating modes.

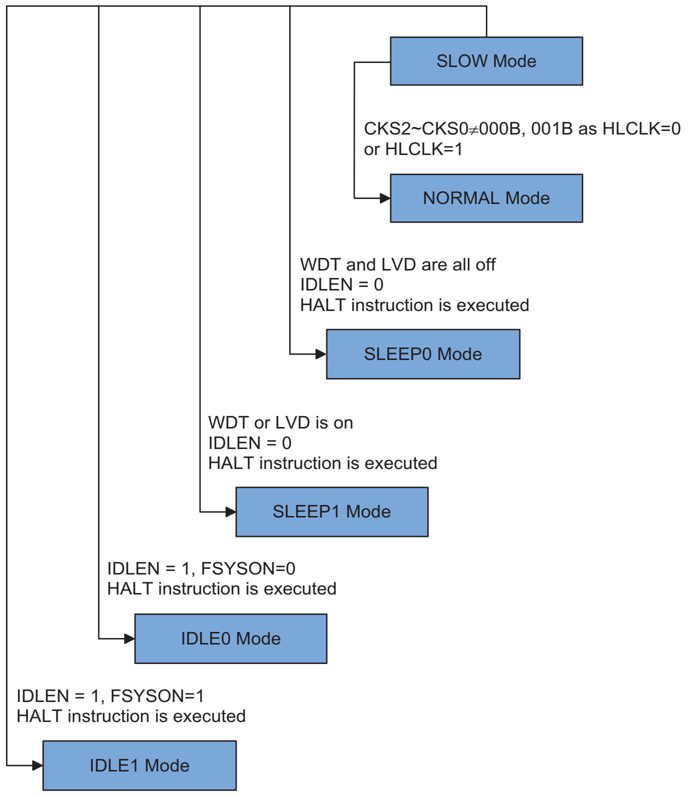


NORMAL Mode to SLOW Mode Switching

When running in the NORMAL Mode, which uses the high speed system oscillator, and therefore consumes more power, the system clock can switch to run in the SLOW Mode by set the HLCLK bit to "0" and set the CKS2~CKS0 bits to "000" or "001" in the SMOD register. This will then use the low speed system oscillator which will consume less power. Users may decide to do this for certain operations which do not require high performance and can subsequently reduce power consumption.

The SLOW Mode is sourced from the LXT or the LIRC oscillators and therefore requires these oscillators to be stable before full mode switching occurs. This is monitored using the LTO bit in the SMOD register.





SLOW Mode to NORMAL Mode Switching

In SLOW Mode the system uses either the LXT or LIRC low speed system oscillator. To switch back to the NORMAL Mode, where the high speed system oscillator is used, the HLCLK bit should be set to "1" or HLCLK bit is "0", but CKS2~CKS0 is set to "010", "011", "100", "101", "110" or "111". As a certain amount of time will be required for the high frequency clock to stabilise, the status of the HTO bit is checked. The amount of time required for high speed system oscillator stabilization depends upon which high speed system oscillator type is used.

Entering the SLEEP0 Mode

There is only one way for the device to enter the SLEEP0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "0" and the WDT and LVD both off. When this instruction is executed under the conditions described above, the following will occur:

- The system clock and the f_{SUB} clock will be stopped and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and stopped as the WDT is disabled.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the SLEEP1 Mode

There is only one way for the device to enter the SLEEP1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "0" and the WDT or LVD on. When this instruction is executed under the conditions described above, the following will occur:

- The system clock will be stopped and the application program will stop at the "HALT" instruction, but the WDT or LVD will remain active with the clock source coming from the f_{SUB} clock.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT is enabled and its clock source is selected to come from the f_{SUB} clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE0 Mode

There is only one way for the device to enter the IDLE0 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "1" and the FSYSON bit in SMOD1 register equal to "0". When this instruction is executed under the conditions described above, the following will occur:

- The system clock will be stopped and the application program will stop at the "HALT" instruction, but the f_{SUB} clock will be on.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT clock source is derived from the f_{SUB} clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Entering the IDLE1 Mode

There is only one way for the device to enter the IDLE1 Mode and that is to execute the "HALT" instruction in the application program with the IDLEN bit in SMOD register equal to "1" and the FSYSON bit in SMOD1 register equal to "1". When this instruction is executed under the conditions described above, the following will occur:

- The system clock and f_{SUB} clock will be on and the application program will stop at the "HALT" instruction.
- The Data Memory contents and registers will maintain their present condition.
- The WDT will be cleared and resume counting as the WDT clock source is derived from the f_{SUB} clock.
- The I/O ports will maintain their present conditions.
- In the status register, the Power Down flag, PDF, will be set and the Watchdog time-out flag, TO, will be cleared.

Standby Current Considerations

As the main reason for entering the SLEEP or IDLE Mode is to keep the current consumption of the device to as low a value as possible, perhaps only in the order of several micro-amps except in the IDLE1 Mode, there are other considerations which must also be taken into account by the circuit designer if the power consumption is to be minimised. Special attention must be made to the I/O pins on the device. All high-impedance input pins must be connected to either a fixed high or low level as any floating input pins could create internal oscillations and result in increased current consumption. This also applies to devices which have different package types, as there may be unbonbed pins. These must either be setup as outputs or if setup as inputs must have pull-high resistors connected.

Care must also be taken with the loads, which are connected to I/O pins, which are setup as outputs. These should be placed in a condition in which minimum current is drawn or connected only to external circuits that do not draw current, such as other CMOS inputs. Also note that additional standby current will also be required if the configuration options have enabled the LXT or LIRC oscillator.

In the IDLE1 Mode the system oscillator is on, if the system oscillator is from the high speed system oscillator, the additional standby current will also be perhaps in the order of several hundred micro-amps.

Wake-up

After the system enters the SLEEP or IDLE Mode, it can be woken up from one of various sources listed as follows:

- An external reset
- An external falling edge on Port A
- A system interrupt
- A WDT overflow

If the device is woken up by a WDT overflow, a Watchdog Timer reset will be initiated. The PDF flag is cleared by a system power-up or executing the clear Watchdog Timer instructions and is set when executing the "HALT" instruction. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the Program Counter and Stack Pointer, the other flags remain in their original status.

Each pin on Port A can be setup using the PAWU register to permit a negative transition on the pin to wake-up the system. When a Port A pin wake-up occurs, the program will resume execution at the instruction following the "HALT" instruction. If the system is woken up by an interrupt, then two possible situations may occur. The first is where the related interrupt is disabled or the interrupt is enabled but the stack is full, in which case the program will resume execution at the instruction following the "HALT" instruction. In this situation, the interrupt which woke-up the device will not be immediately serviced, but will rather be serviced later when the related interrupt is finally enabled or when a stack level becomes free. The other situation is where the related interrupt is enabled and the stack is not full, in which case the regular interrupt response takes place. If an interrupt request flag is set high before entering the SLEEP or IDLE Mode, the wake-up function of the related interrupt will be disabled.

Programming Considerations

The HXT and LXT oscillators both use the same SST counter. For example, if the system is woken up from the SLEEP0 Mode and both the HXT and LXT oscillators need to start-up from an off state. The LXT oscillator uses the SST counter after HXT oscillator has finished its SST period.

- If the device is woken up from the SLEEP0 Mode to the NORMAL Mode, the high speed system oscillator needs an SST period. The device will execute first instruction after HTO is "1". At this time, the LXT oscillator may not be stable if f_{SUB} is from LXT oscillator. The same situation occurs in the power-on state. The LXT oscillator is not ready yet when the first instruction is executed.
- If the device is woken up from the SLEEP1 Mode to NORMAL Mode, and the system clock source is from HXT oscillator and FSTEN is "1", the system clock can be switched to the LXT or LIRC oscillator after wake up.
- There are peripheral functions, such as Time Base and TMs, for which the f_{SYS} is used. If the system clock source is switched from f_H to f_{SUB}, the clock source to the peripheral functions mentioned above will change accordingly.
- The f_{SUB} on/off condition depends upon whether the WDT or LVD is enabled or both disabled as the clock source is selected from f_{SUB}.

Watchdog Timer

The Watchdog Timer is provided to prevent program malfunctions or sequences from jumping to unknown locations, due to certain uncontrollable external events such as electrical noise.

Watchdog Timer Clock Source

The Watchdog Timer clock source is provided by the internal clock which is supplied by the f_{SUB} clock. The f_{SUB} clock can be sourced from either the LXT or LIRC oscillator selected by a configuration option. The LIRC internal oscillator has an approximate frequency of 32kHz and this specified internal clock period can vary with V_{DD}, temperature and process variations. The LXT oscillator is supplied by an external 32.768kHz crystal. The Watchdog Timer source clock is then subdivided by a ratio of 2^8 to 2^{18} to give longer timeouts, the actual value being chosen using the WS2~WS0 bits in the WDTC register.

Watchdog Timer Control Register

A single register, WDTC, controls the required time-out period as well as the enable/disable operation. This register controls the overall operation of the Watchdog Timer.

WDTC Register

Bit	7	6	5	4	3	2	1	0
Name	WE4	WE3	WE2	WE1	WE0	WS2	WS1	WS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	1	0	1	0	0	1	1

Bit 7~3 **WE4~WE0:** WDT function enable control
10101: Disabled
01010: Enabled
Other Values: Reset MCU
If these bits are changed due to adverse environmental conditions, the microcontroller will be reset. The reset operation will be activated after 2~3 f_{SUB} clock cycles and the WRF bit in the SMOD1 register will be set to 1.

Bit 2~0 **WS2~WS0:** Select WDT Timeout Period
000: $2^8/f_S$
001: $2^{10}/f_S$
010: $2^{12}/f_S$
011: $2^{14}/f_S$
100: $2^{15}/f_S$
101: $2^{16}/f_S$
110: $2^{17}/f_S$
111: $2^{18}/f_S$
These three bits determine the division ratio of the Watchdog Timer source clock, which in turn determines the timeout period.

SMOD1 Register

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	RSTF	LVRF	LRF	WRF
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	x	0	0

"x" unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
Described elsewhere.

Bit 6~4 Unimplemented, read as "0"

Bit 3 **RSTF:** RSTC Control register software reset flag
Described elsewhere.

Bit 2 **LVRF:** LVR function reset flag
Described elsewhere.

Bit 1 **LRF:** LVR Control register software reset flag
Described elsewhere.

bit 0 **WRF:** WDT Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 by the WDT Control register software reset and cleared by the application program. Note that this bit can only be cleared to 0 by the application program.

Watchdog Timer Operation

The Watchdog Timer operates by providing a device reset when its timer overflows. This means that in the application program and during normal operation the user has to strategically clear the Watchdog Timer before it overflows to prevent the Watchdog Timer from executing a reset. This is done using the clear watchdog instructions. If the program malfunctions for whatever reason, jumps to an unknown location, or enters an endless loop, these clear instructions will not be executed in the correct manner, in which case the Watchdog Timer will overflow and reset the device. With regard to the Watchdog Timer enable/disable function, there are five bits, WE4~WE0, in the WDTC register to offer the additional enable/disable control and reset control of the Watchdog Timer. The WDT function will be disabled when the WE4~WE0 bits are set to a value of 10101B while the WDT function will be enabled if the WE4~WE0 bits are equal to 01010B. If the WE4~WE0 bits are set to any other values, other than 01010B and 10101B, it will reset the device after 2~3 f_{SUB} clock cycles. After power on these bits will have a value of 01010B.

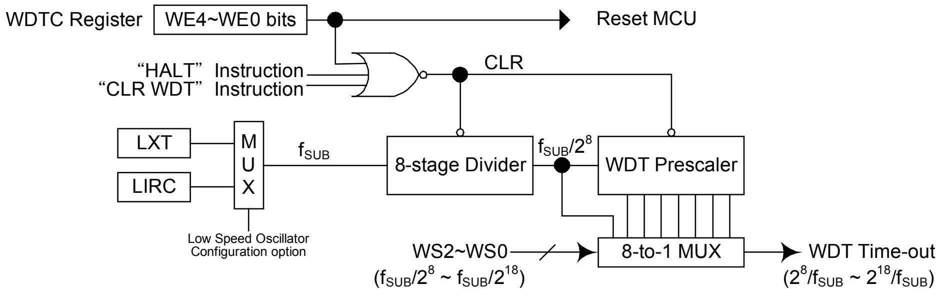
WE4~WE0 Bits	WDT Function
10101B	Disable
01010B	Enable
Any other value	Reset MCU

Watchdog Timer Enable/Disable Control

Under normal program operation, a Watchdog Timer time-out will initialise a device reset and set the status bit TO. However, if the system is in the SLEEP or IDLE Mode, when a Watchdog Timer time-out occurs, the TO bit in the status register will be set and only the Program Counter and Stack Pointer will be reset. Three methods can be adopted to clear the contents of the Watchdog Timer. The first is a WDT reset, which means a certain value except 01010B and 10101B written into the WE4~WE0 field, the second is using the Watchdog Timer software clear instruction and the third is via a HALT instruction.

There is only one method of using software instruction to clear the Watchdog Timer. That is to use the single "CLR WDT" instruction to clear the WDT contents.

The maximum time out period is when the 2^{18} division ratio is selected. As an example, with a 32kHz LIRC oscillator as its source clock, this will give a maximum watchdog period of around 8 second for the 2^{18} division ratio, and a minimum timeout of 7.8ms for the 2^{8} division ration.



Watchdog Timer

Reset and Initialisation

A reset function is a fundamental part of any microcontroller ensuring that the device can be set to some predetermined condition irrespective of outside parameters. The most important reset condition is after power is first applied to the microcontroller. In this case, internal circuitry will ensure that the microcontroller, after a short delay, will be in a well defined state and ready to execute the first program instruction. After this power-on reset, certain important internal registers will be set to defined states before the program commences. One of these registers is the Program Counter, which will be reset to zero forcing the microcontroller to begin program execution from the lowest Program Memory address.

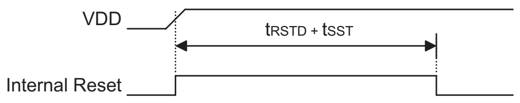
Another type of reset is when the Watchdog Timer overflows and resets the MCU. All types of reset operations result in different register conditions being setup. Another reset exists in the form of a Low Voltage Reset, LVR, where a full reset, is implemented in situations where the power supply voltage falls below a certain threshold.

Reset Functions

There are four ways in which a reset can occur.

Power-on Reset

The most fundamental and unavoidable reset is the one that occurs after power is first applied to the. As well as ensuring that the Program Memory begins execution from the first memory address, a power-on reset also ensures that certain other registers are preset to known conditions. All the I/O port and port control registers will power up in a high condition ensuring that all pins will be first set to inputs.

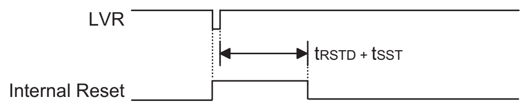


Power-On Reset Timing Chart

Note: t_{RSTD} is power-on delay, typical time=50ms

Low Voltage Reset – LVR

The microcontroller contains a low voltage reset circuit in order to monitor the supply voltage of the device. The LVR function is always enabled with a specific LVR voltage, V_{LVR}. If the supply voltage of the device drops to within a range of 0.9V~V_{LVR} such as might occur when changing the battery, the LVR will automatically reset the device internally and the LVRF bit in the SMOD1 register will also be set to 1. For a valid LVR signal, a low supply voltage, i.e., a voltage in the range between 0.9V~V_{LVR} must exist for a time greater than that specified by t_{LVR} in the A.C. characteristics. If the low supply voltage state does not exceed this value, the LVR will ignore the low supply voltage and will not perform a reset function. The actual V_{LVR} value can be selected by the LVS bits in the LVRC register. If the LVS7~LVS0 bits have any other value, which may perhaps occur due to adverse environmental conditions such as noise, the LVR will reset the device after 2~3 f_{SUB} clock cycles. When this happens, the LRF bit in the SMOD1 register will be set to 1. After power on the register will have the value of 01010101B. Note that the LVR function will be automatically disabled when the device enters the power down mode.



Note: t_{RSTD} is power-on delay, typical time=50ms

Low Voltage Reset Timing Chart

- **LVRC Register**

Bit	7	6	5	4	3	2	1	0
Name	LVS7	LVS6	LVS5	LVS4	LVS3	LVS2	LVS1	LVS0
R/W	R/W	R/W	R/W	R/W	R/w	R/w	R/W	R/W
POR	0	1	0	1	0	1	0	1

Bit 7~0 **LVS7~LVS0:** LVR voltage select
01010101: 2.1V
00110011: 2.55V
10011001: 3.15V
10101010: 3.8V
Any other values: Generates MCU reset – register is reset to POR value

When an actual low voltage condition occurs, as specified by one of the four defined LVR voltage values above, an MCU reset will be generated. The reset operation will be activated after 2~3 f_{SUB} clock cycles. In this situation the register contents will remain the same after such a reset occurs.

Any register value, other than the four defined register values above, will also result in the generation of an MCU reset. The reset operation will be activated after 2~3 f_{SUB} clock cycles. However in this situation the register contents will be reset to the POR value.

• **SMOD1 Register**

Bit	7	6	5	4	3	2	1	0
Name	FSYSON	—	—	—	RSTF	LVRF	LRF	WRF
R/W	R/W	—	—	—	R/W	R/W	R/W	R/W
POR	0	—	—	—	0	x	0	0

"x" unknown

Bit 7 **FSYSON:** f_{SYS} Control in IDLE Mode
Described elsewhere.

Bit 6~4 Unimplemented, read as "0"

Bit 3 **RSTF:** RSTC Control register software reset flag
Described elsewhere.

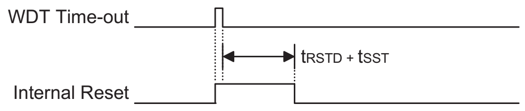
Bit 2 **LVRF:** LVR function reset flag
0: Not occurred
1: Occurred
This bit is set to 1 when a specific Low Voltage Reset situation condition occurs. This bit can only be cleared to 0 by the application program.

Bit 1 **LRF:** LVR Control register software reset flag
0: Not occurred
1: Occurred
This bit is set to 1 if the LVRC register contains any non defined LVR voltage register values. This in effect acts like a software-reset function. This bit can only be cleared to 0 by the application program.

bit 0 **WRF:** WDT Control register software reset flag
Described elsewhere.

Watchdog Time-out Reset during Normal Operation

The Watchdog time-out Reset during normal operation is the same as a hardware $\overline{RES}$ pin reset except that the Watchdog time-out flag TO will be set to "1".

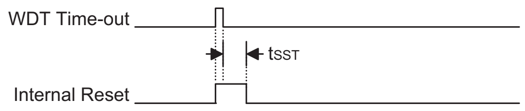


WDT Time-out Reset during Normal Operation Timing Chart

Note: t_{RSTD} is power-on delay, typical time=16.7ms

Watchdog Time-out Reset during SLEEP or IDLE Mode

The Watchdog time-out Reset during SLEEP or IDLE Mode is a little different from other kinds of reset. Most of the conditions remain unchanged except that the Program Counter and the Stack Pointer will be cleared to "0" and the TO flag will be set to "1". Refer to the A.C. Characteristics for t_{SST} details.



WDT Time-out Reset during SLEEP or IDLE Timing Chart

Note: The t_{SST} is 15~16 clock cycles if the system clock source is provided by ERC or HIRC. The t_{SST} is 1024 clock for HXT or LXT. The t_{SST} is 1~2 clock for LIRC.

Reset Initial Conditions

The different types of reset described affect the reset flags in different ways. These flags, known as PDF and TO are located in the status register and are controlled by various microcontroller operations, such as the SLEEP or IDLE Mode function or Watchdog Timer. The reset flags are shown in the table:

TO	PDF	RESET Conditions
0	0	Power-on reset
u	u	LVR reset during NORMAL or SLOW Mode operation
1	u	WDT time-out reset during NORMAL or SLOW Mode operation
1	1	WDT time-out reset during IDLE or SLEEP Mode operation

"u" stands for unchanged

The following table indicates the way in which the various components of the microcontroller are affected after a power-on reset occurs.

Item	Condition After RESET
Program Counter	Reset to zero
Interrupts	All interrupts will be disabled
WDT, Time Base	Clear after reset, WDT begins counting
Timer Modules	Timer Modules will be turned off
Input/Output Ports	I/O ports will be setup as inputs
Stack Pointer	Stack Pointer will point to the top of the stack

The different kinds of resets all affect the internal registers of the microcontroller in different ways. To ensure reliable continuation of normal program execution after a reset occurs, it is important to know what condition the microcontroller is in after a particular reset occurs. The following table describes how each type of reset affects the microcontroller internal registers.

Register	Power-on Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
IAR0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
IAR1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
BP	---- ---0	---- ---0	---- ---0	---- ---u
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
PCL	0000 0000	0000 0000	0000 0000	0000 0000
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBHP	---- xxxx	---- uuuu	---- uuuu	---- uuuu
STATUS	--00 xxxx	--uu uuuu	--1u uuuu	--11 uuuu
SMOD	0000 0011	0000 0011	0000 0011	uuuu uuuu
LVDC	--00 -000	--00 -000	--00 -000	--uu -uuu
INTEG	---- 0000	---- 0000	---- 0000	---- uuuu
WDTC	0101 0011	0101 0011	0101 0011	uuuu uuuu
TBC	0011 0111	0011 0111	0011 0111	uuuu uuuu
INTC0	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
SMOD1	0--- xx00	0--- xx00	0--- uuuu	u--- uuuu

Register	Power-on Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
LVRC	0101 0101	uuuu uuuu	0101 0101	uuuu uuuu
MFI0	0000 0000	0000 0000	0000 0000	uuuu uuuu
MFI1	-000 -000	-000 -000	-000 -000	-uuu -uuu
MFI2	--00 --00	--00 --00	--00 --00	--uu --uu
INTC2	--00 --00	--00 --00	--00 --00	--uu --uu
PAWU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PA	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCPU	0000 0000	0000 0000	0000 0000	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	uuuu uuuu
PDPU	---- --00	---- --00	---- --00	---- --uu
PD	---- --11	---- --11	---- --11	---- --uu
PDC	---- --11	---- --11	---- --11	---- --uu
RSTC	0101 0101	0101 0101	0101 0101	uuuu uuuu
PXFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PXFS1	--00 0000	--00 0000	--00 0000	--uu uuuu
PAFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PAFS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PBFS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS0	0000 0000	0000 0000	0000 0000	uuuu uuuu
PCFS1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PDFS0	---- 0000	---- 0000	---- 0000	---- uuuu
CTMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMDH	---- --00	---- --00	---- --00	---- --uu
CTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
CTMAH	---- --00	---- --00	---- --00	---- --uu
EEA	--00 0000	--00 0000	--00 0000	--uu uuuu
EED	0000 0000	0000 0000	0000 0000	uuuu uuuu
EEC	---- 0000	---- 0000	---- 0000	---- uuuu
ETMC0	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMC2	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMDH	---- --00	---- --00	---- --00	---- --uu

Register	Power-on Reset	LVR Reset	WDT Time-out (Normal Operation)	WDT Time-out (HALT)
ETMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMAH	---- --00	---- --00	---- --00	---- --uu
ETMBL	0000 0000	0000 0000	0000 0000	uuuu uuuu
ETMBH	---- --00	---- --00	---- --00	---- --uu
PTMC0	0000 0---	0000 0---	0000 0---	uuuu u---
PTMC1	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMDH	---- --00	---- --00	---- --00	---- --uu
PTMAL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMAH	---- --00	---- --00	---- --00	---- --uu
PTMRPL	0000 0000	0000 0000	0000 0000	uuuu uuuu
PTMRPH	---- --00	---- --00	---- --00	---- --uu
IICC0	---- 000-	---- 000-	---- 000-	---- uuu-
IICC1	1000 0001	1000 0001	1000 0001	uuuu uuuu
IICD	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx
IICA	0000 000-	0000 000-	0000 000-	uuuu uuu-
I2CTOC	0000 0000	0000 0000	0000 0000	uuuu uuuu
SLCDC	-000 ----	-000 ----	-000 ----	-uuu ----
USR	0000 1011	0000 1011	0000 1011	uuuu uuuu
UCR1	0000 00x0	0000 00x0	0000 00x0	uuuu uuuu
UCR2	0000 0000	0000 0000	0000 0000	uuuu uuuu
BRG	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TXR_RXR	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu

Note: "-" not implement
"u" means "unchanged"
"x" means "unknown"

Input/Output Ports

Holtek microcontrollers offer considerable flexibility on their I/O ports. With the input or output designation of every pin fully under user program control, pull-high selections for all ports and wake-up selections on certain pins, the user is provided with an I/O structure to meet the needs of a wide range of application possibilities.

The device provides bidirectional input/output lines labeled with port names PA~PD. These I/O ports are mapped to the RAM Data Memory with specific addresses as shown in the Special Purpose Data Memory table. All of these I/O ports can be used for input and output operations. For input operation, these ports are non-latching, which means the inputs must be ready at the T2 rising edge of instruction "MOV A, [m]", where m denotes the port address. For output operation, all the data is latched and remains unchanged until the output latch is rewritten.

I/O Port Register List

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAWU	PAWU7	PAWU 6	PAWU5	PAWU4	PAWU3	PAWU2	PAWU1	PAWU0
PAPU	PAPU7	PAPU6	PAPU5	PAPU4	PAPU3	PAPU2	PAPU1	PAPU0
PA	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
PAC	PAC7	PAC6	PAC5	PAC4	PAC3	PAC2	PAC1	PAC0
PBPU	PBPU7	PBPU6	PBPU5	PBPU4	PBPU3	PBPU2	PBPU1	PBPU0
PB	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
PBC	PBC7	PBC6	PBC5	PBC4	PBC3	PBC2	PBC1	PBC0
PCPU	PCPU7	PCPU6	PCPU5	PCPU4	PCPU3	PCPU2	PCPU1	PCPU0
PC	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
PCC	PCC7	PCC6	PCC4	PCC4	PCC3	PCC2	PCC1	PCC0
PDPU	—	—	—	—	—	—	PDPU1	PDPU0
PD	—	—	—	—	—	—	PD1	PD0
PDC	—	—	—	—	—	—	PDC1	PDC0

"—" Unimplemented, read as "0"

PAWUn: PA wake-up function control
0: Disable
1: Enable

PAn/PBn/PCn/PDn: I/O Data bit
0: Data 0
1: Data 1

PACn/PBCn/PCCn/PDCn: I/O type selection
0: Output
1: Input

PAPUn/PBPUn/PCPUn/PDPUn: Pull-high function control
0: Disable
1: Enable

Pull-high Resistors

Many product applications require pull-high resistors for their switch inputs usually requiring the use of an external resistor. To eliminate the need for these external resistors, all I/O pins, when configured as an input have the capability of being connected to an internal pull-high resistor. These pull-high resistors are selected using registers PAPU~PDPU, and are implemented using weak PMOS transistors.

Port A Wake-up

The HALT instruction forces the microcontroller into the SLEEP or IDLE Mode which preserves power, a feature that is important for battery and other low-power applications. Various methods exist to wake-up the microcontroller, one of which is to change the logic condition on one of the Port A pins from high to low. This function is especially suitable for applications that can be woken up via external switches. Each pin on Port A can be selected individually to have this wake-up feature using the PAWU register.

I/O Port Control Registers

Each Port has its own control register, known as PAC~PDC, which controls the input/output configuration. With this control register, each I/O pin with or without pull-high resistors can be reconfigured dynamically under software control. For the I/O pin to function as an input, the corresponding bit of the control register must be written as a "1". This will then allow the logic state of the input pin to be directly read by instructions. When the corresponding bit of the control register is written as a "0", the I/O pin will be setup as a CMOS output. If the pin is currently setup as an output, instructions can still be used to read the output register.

However, it should be noted that the program will in fact only read the status of the output data latch and not the actual logic status of the output pin.

Pin-shared Functions

The flexibility of the microcontroller range is greatly enhanced by the use of pins that have more than one function. Limited numbers of pins can force serious design constraints on designers but by supplying pins with multi-functions, many of these difficulties can be overcome. For these pins, the chosen function of the multi-function I/O pins is selected by a series of regiters via the application program control.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PAFS0	PAFS07	PAFS06	PAFS05	PAFS04	PAFS03	PAFS02	PAFS01	PAFS00
PAFS1	PAFS17	PAFS16	PAFS15	PAFS14	PAFS13	PAFS12	PAFS11	PAFS10
PBFS0	PBFS07	PBFS06	PBFS05	PBFS04	PBFS03	PBFS02	PBFS01	PBFS00
PBFS1	PBFS17	PBFS16	PBFS15	PBFS14	PBFS13	PBFS12	PBFS11	PBFS10
PCFS0	PCFS07	PCFS06	PCFS05	PCFS04	PCFS03	PCFS02	PCFS01	PCFS00
PCFS1	PCFS17	PCFS16	PCFS15	PCFS14	PCFS13	PCFS12	PCFS11	PCFS10
PDFS0	—	—	—	—	PDFS03	PDFS02	PDFS01	PDFS00

Pin-shared Function Selection Register List

PAFS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PAFS07	PAFS06	PAFS05	PAFS04	PAFS03	PAFS02	PAFS01	PAFS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

"x": don't care

Bit 7~6 **PAFS07~PAFS06:** Port A 3 Function Selection
00: PA3/LED1
01: PA3/LED1
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PAFS05~PAFS04:** Port A 2 Function Selection
xx: PA2

Bit 3~2 **PAFS03~PAFS02:** Port A 1 Function Selection
00: PA1/LED0
01: PA1/LED0
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PAFS01~PAFS00:** Port A 0 Function Selection
xx: PA0

PAFS1 Register

Bit	7	6	5	4	3	2	1	0
Name	PAFS17	PAFS16	PAFS15	PAFS14	PAFS13	PAFS12	PAFS11	PAFS10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PAFS17~PAFS16:** Port A 7 Function Selection
00: PA7/LED5
01: RX
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PAFS15~PAFS14:** Port A 6 Function Selection
00: PA6/LED4
01: TX
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 3~2 **PAFS13~PAFS12:** Port A 5 Function Selection
00: PA5/LED3
01: ETPA
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PAFS11~PAFS10:** Port A 4 Function Selection
00: PA4/LED2
01: CTP
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

PBFS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PBFS07	PBFS06	PBFS05	PBFS04	PBFS03	PBFS02	PBFS01	PBFS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PBFS07~PBFS06:** Port B 3 Function Selection
00: PB3/LED9
01: PTP
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PBFS05~PBFS04:** Port B 2 Function Selection
00: PB2/LED8
01: ETPB
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 3~2 **PBFS03~PBFS02:** Port B 1 Function Selection
00: PB1/LED7
01: SCL
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PBFS01~PBFS00:** Port B 0 Function Selection
00: PB0/LED6
01: SDA
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

PBFS1 Register

Bit	7	6	5	4	3	2	1	0
Name	PBFS17	PBFS16	PBFS15	PBFS14	PBFS13	PBFS12	PBFS11	PBFS10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PBFS17~PBFS16:** Port B 7 Function Selection
00: PB7/LED13
01: PB7/LED13
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PBFS15~PBFS14:** Port B 6 Function Selection
00: PB6/LED12
01: PB6/LED12
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 3~2 **PBFS13~PBFS12:** Port B 5 Function Selection
00: PB5/LED11
01: PB5/LED11
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PBFS11~PBFS10:** Port B 4 Function Selection
00: PB4/LED10
01: PB4/LED10
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

PCFS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PCFS07	PCFS06	PCFS05	PCFS04	PCFS03	PCFS02	PCFS01	PCFS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PCFS07~PCFS06:** Port C 3 Function Selection
00: PC3/LED17
01: PC3/LED17
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PCFS05~PCFS04:** Port C 2 Function Selection
00: PC2/LED16
01: PC2/LED16
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 3~2 **PCFS03~PCFS02:** Port C 1 Function Selection
00: PC1/LED15
01: PC1/LED15
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PCFS01~PCFS00:** Port C 0 Function Selection
00: PC0/LED14
01: PC0/LED14
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

PCFS1 Register

Bit	7	6	5	4	3	2	1	0
Name	PCFS17	PCFS16	PCFS15	PCFS14	PCFS13	PCFS12	PCFS11	PCFS10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PCFS17~PCFS16:** Port C 7 Function Selection
00: PC7/LED21
01: PC7/LED21
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 5~4 **PCFS15~PCFS14:** Port C 6 Function Selection
00: PC6/LED20
01: PC6/LED20
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 3~2 **PCFS13~PCFS12:** Port C 5 Function Selection
00: PC5/LED19
01: PC5/LED19
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PCFS11~PCFS10:** Port C 4 Function Selection
00: PC4/LED18
01: PC4/LED18
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

PDFS0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	PDFS03	PDFS02	PDFS01	PDFS00
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **PDFS03~PDFS02:** Port D 1 Function Selection
00: PD1/LED23
01: PD1/LED23
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

Bit 1~0 **PDFS01~PDFS00:** Port D 0 Function Selection
00: PD0/LED22
01: PD0/LED22
10: SEG_L (1/3*V_{DD})
11: SEG_H (2/3*V_{DD})

I/O Pin Structures

The accompanying diagrams illustrate the internal structures of some generic I/O pin types. As the exact logical construction of the I/O pin will differ from these drawings, they are supplied as a guide only to assist with the functional understanding of the I/O pins. The wide range of pin-shared structures does not permit all types to be shown.

I/O Source Current Selection

The device provides 24 I/O lines with the programmable source current functions. There are four levels source current from level 0 to level 3 selected by the PXPS1 or PXPS0 register for various applications such as a LED matrix. Users can refer to the D.C. characteristics section to select the desired source current level for specific applications.

Register Name	Bit							
	7	6	5	4	3	2	1	0
PXFS0	PXPS07	PXPS06	PXPS05	PXPS04	PXPS03	PXPS02	PXPS01	PXPS00
PXFS1	—	—	PXPS15	PXPS14	PXPS13	PXPS12	PXPS11	PXPS10

Source Current Selection Register List

PXPS0 Register

Bit	7	6	5	4	3	2	1	0
Name	PXPS07	PXPS06	PXPS05	PXPS04	PXPS03	PXPS02	PXPS01	PXPS00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PXPS07~PXPS06:** PB7~PB4 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

Bit 5~4 **PXPS05~PXPS04:** PB3~PB0 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

Bit 3~2 **PXPS03~PXPS02:** PA7~PA4 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

Bit 1~0 **PXPS01~PXPS00:** PA3 and PA0 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

PXPS1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	PXPS15	PXPS14	PXPS13	PXPS12	PXPS11	PXPS10
R/W	—	—	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	—	0	0	0	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5~4 **PXPS15~PXPS14:** PD1~PD0 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

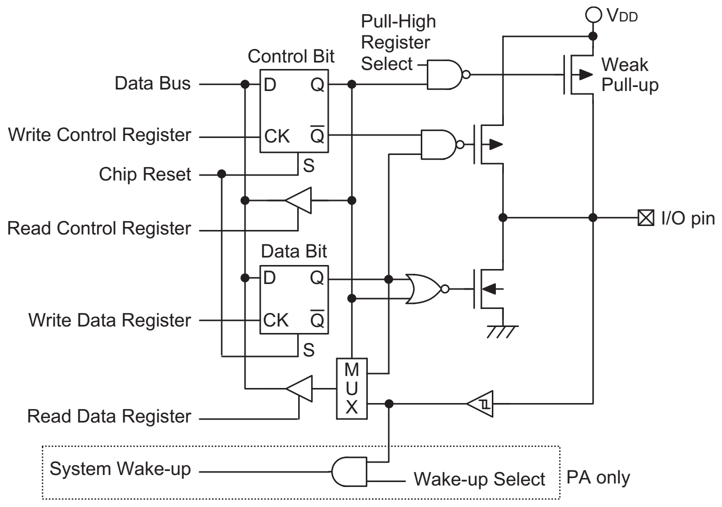
Bit 3~2 **PXPS13~PXPS12:** PC7~PC4 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

Bit 1~0 **PXPS11~PXPS10:** PC3 ~ PC0 Source Current Selection
00: Source current level 3 (max.)
01: Source current level 2
10: Source current level 1
11: Source current level 0 (min.)

Programming Considerations

Within the user program, one of the things first to consider is port initialisation. After a reset, all of the I/O data and port control registers will be set to high. This means that all I/O pins will be defaulted to an input state, the level of which depends on the other connected circuitry and whether pull-high selections have been chosen. If the port control registers are then programmed to setup some pins as outputs, these output pins will have an initial high output value unless the associated port data registers are first programmed. Selecting which pins are inputs and which are outputs can be achieved byte-wide by loading the correct values into the appropriate port control register or by programming individual bits in the port control register using the "SET [m].i" and "CLR [m].i" instructions. Note that when using these bit control instructions, a read-modify-write operation takes place. The microcontroller must first read in the data on the entire port, modify it to the required new bit values and then rewrite this data back to the output ports.

Port A has the additional capability of providing wake-up functions. When the device is in the SLEEP or IDLE Mode, various methods are available to wake the device up. One of these is a high to low transition of any of the Port A pins. Single or multiple pins on Port A can be setup to have this function.



Generic Input/Output Structure

Timer Modules – TM

One of the most fundamental functions in any microcontroller device is the ability to control and measure time. To implement time related functions the device includes several Timer Modules, abbreviated to the name TM. The TMs are multi-purpose timing units and serve to provide operations such as Timer/Counter, Input Capture, Compare Match Output and Single Pulse Output as well as being the functional unit for the generation of PWM signals. Each of the TMs has either two or three individual interrupts. The addition of input and output pins for each TM ensures that users are provided with timing units with a wide and flexible range of features.

The common features of the different TM types are described here with more detailed information provided in the individual Compact, Periodic and Enhanced TM sections.

Introduction

The device contains three TMs with each TM having a reference name of CTM, PTM, and ETM. Each individual TM can be named as Compact Type TM, Periodic Type TM or Enhanced Type TM. Although similar in nature, the different TM types vary in their feature complexity. The common features to all of the Compact, Periodic and Enhanced TMs will be described in this section and the detailed operation regarding each of the TM types will be described in separate sections. The main features and differences between the three types of TMs are summarised in the accompanying table.

TM Function	CTM	PTM	ETM
Timer/Counter	√	√	√
I/P Capture	—	√	√
Compare Match Output	√	√	√
PWM Channels	1	1	2
Single Pulse Output	—	1	2
PWM Alignment	Edge	Edge	Edge & Centre
PWM Adjustment Period & Duty	Duty or Period	Duty or Period	Duty or Period

TM Function Summary

TM Operation

The three different types of TM offer a diverse range of functions, from simple timing operations to PWM signal generation. The key to understanding how the TM operates is to see it in terms of a free running counter whose value is then compared with the value of pre-programmed internal comparators. When the free running counter has the same value as the pre-programmed comparator, known as a compare match situation, a TM interrupt signal will be generated which can clear the counter and perhaps also change the condition of the TM output pin. The internal TM counter is driven by a user selectable clock source, which can be an internal clock or an external pin.

TM Clock Source

The clock source which drives the main counter in each TM can originate from various sources. The selection of the required clock source is implemented using the xTCK2~xTCK0 bits in the xTM control registers where x can be C, P or E, which stands for the different types of TMs. The clock source can be a ratio of the system clock f_{SYS} or the internal high clock f_H, the f_{SUB} clock source or the external xTCK pin. The xTCK pin clock source is used to allow an external signal to drive the TM as an external clock source or for event counting.

TM Interrupts

The Compact and Periodic type TMs each have two internal interrupts, one for each of the internal comparator A or comparator P, which generate a TM interrupt when a compare match condition occurs. As the Enhanced type TM has three internal comparators and comparator A or comparator B or comparator P compare match functions, it consequently has three internal interrupts. When a TM interrupt is generated it can be used to clear the counter and also to change the state of the TM output pin.

TM External Pins

Each of the TMs, irrespective of what type, has one TM input pin, with the label xTCK. The TM input pin, is essentially a clock source for the TM and is selected using the xTCK2~xTCK0 bits in the xTMC0 register. This external TM input pin allows an external clock source to drive the internal TM. This external TM input pin is shared with other functions but will be connected to the internal TM if selected using the xTCK2~xTCK0 bits. The TM input pin can be chosen to have either a rising or falling active edge.

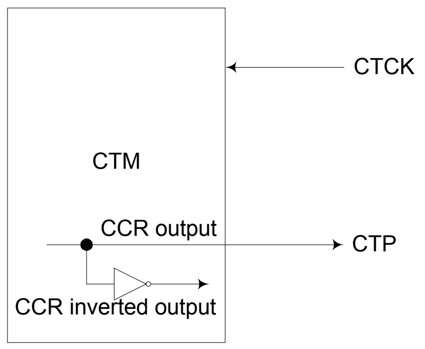
The TMs each have one or more output pins with the label xTP. When the TM is in the Compare Match Output Mode, these pins can be controlled by the TM to switch to a high or low level or to toggle when a compare match situation occurs. The external xTP output pin is also the pin where the TM generates the PWM output waveform. As the TM output pins are pin-shared with other function, the TM output function must first be setup using the corresponding pin shared control registers. The corresponding bits in these control registers determine if its associated pin is to be used as an external TM output pin or if it is to have another function.

Pin / TM Type	Input	Output	Control Registers
CTM	CTCK	CTP	PAFS1
PTM	PTCK, PTPI	PTP	PBFS0
ETM	ETCK, ETPIA, ETPIB	ETPA, ETPB	PAFS1, PBFS0

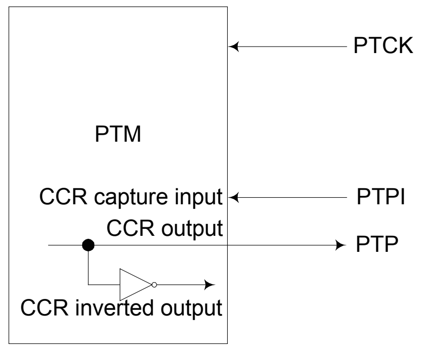
TM Input/Output Pins

TM Input/Output Pin Control Registers

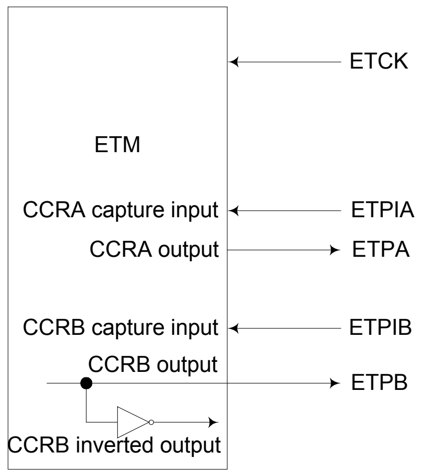
Selecting to have a TM input/output or whether to retain its other shared function is implemented by configuring the corresponding pin-shared control register. The detail definitions of the pin-shared control registers are described in the "Pin-shared Function" section in the I/O chapter.



CTM Function Pin Control Block Diagram



PTM Function Pin Control Block Diagram

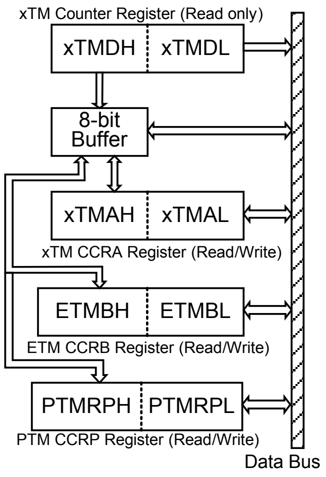


ETM Function Pin Control Block Diagram

Programming Considerations

The TM Counter Registers and the Capture/Compare CCRA and CCRB registers together with the CCRP registers, being 10-bit, all have a low and high byte structure. The high bytes can be directly accessed, but as the low bytes can only be accessed via an internal 8-bit buffer, reading or writing to these register pairs must be carried out in a specific way. The important point to note is that data transfer to and from the 8-bit buffer and its related low byte only takes place when a write or read operation to its corresponding high byte is executed.

As the CCRA, CCRB and CCRP registers are implemented in the way shown in the following diagram and accessing these register pairs is carried out in a specific way described above, it is recommended to use the "MOV" instruction to access the CCRA , CCRB and CCRP low byte registers, named xTMAL, ETMBL and PTMRPL, using the following access procedures. Accessing the CCRA, CCRB or CCRP low byte registers without following these access procedures will result in unpredictable values.


xTM Counter Register (Read only)
xTMDH
xTMDL
8-bit Buffer
xTMAH
xTMAL
xTM CCRA Register (Read/Write)
ETMBH
ETMBL
ETM CCRB Register (Read/Write)
PTMRPH
PTMRPL
PTM CCRP Register (Read/Write)
Data Bus

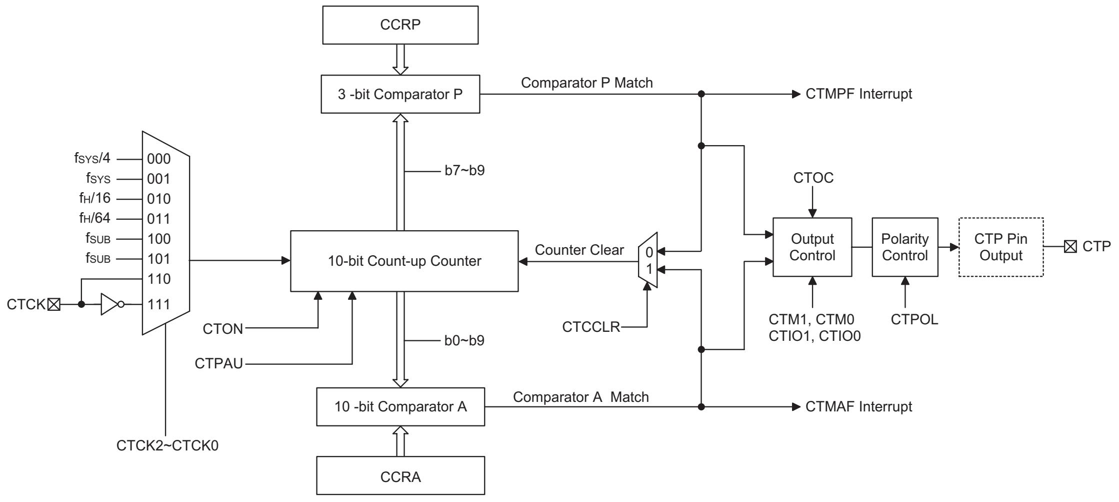
The following steps show the read and write procedures:

- Writing Data to CCRA, CCRB or CCRP
  - ♦ Step 1. Write data to Low Byte xTMAL, ETMBL or PTMRPL
    - – Note that here data is only written to the 8-bit buffer.
  - ♦ Step 2. Write data to High Byte xTMAH, ETMBH or PTMRPH
    - – Here data is written directly to the high byte registers and simultaneously data is latched from the 8-bit buffer to the Low Byte registers.
- Reading Data from the Counter Registers and CCRA, CCRB or CCRP
  - ♦ Step 1. Read data from the High Byte xTMDH, xTMAH, ETMBH or PTMRPH
    - – Here data is read directly from the High Byte registers and simultaneously data is latched from the Low Byte register into the 8-bit buffer.
  - ♦ Step 2. Read data from the Low Byte xTMDL, xTMAL, ETMBL or PTMRPL
    - – This step reads data from the 8-bit buffer.

Compact Type TM – CTM

Although the simplest form of the three TM types, the Compact TM type still contains three operating modes, which are Compare Match Output, Timer/Event Counter and PWM Output modes. The Compact TM can also be controlled with an external input pin and can drive one external output pins.

TM Type	TM Input Pin	TM Output Pin
10-bit CTM	CTCK	CTP



Compact Type TM Block Diagram (n=0)

Compact TM Operation

At its core is a 10-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP is three bits wide whose value is compared with the highest three bits in the counter while the CCRA is the ten bits and therefore compares with all counter bits.

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the CTON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Compact Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.

Compact Type TM Register Description

Overall operation of the Compact TM is controlled using six registers. A read only register pair exists to store the internal counter 10-bit value, while a read/write register pair exists to store the internal 10-bit CCRA value. The remaining two registers are control registers which setup the different operating and control modes as well as the three CCRP bits.

Register Name	Bit							
	7	6	5	4	3	2	1	0
CTMC0	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
CTMC1	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
CTMDL	D7	D6	D5	D4	D3	D2	D1	D0
CTMDH	—	—	—	—	—	—	D9	D8
CTMAL	D7	D6	D5	D4	D3	D2	D1	D0
CTMAH	—	—	—	—	—	—	D9	D8

Compact TM Register List (n=0)

CTMDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **CTMDL**: CTM Counter Low Byte Register bit 7 ~ bit 0
CTM 10-bit Counter bit 7 ~ bit 0

CTMDH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **CTMDH**: CTM Counter High Byte Register bit 1 ~ bit 0
CTM 10-bit Counter bit 9 ~ bit 8

CTMAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **CTMAL**: CTM CCRA Low Byte Register bit 7 ~ bit 0
CTM 10-bit CCRA bit 7 ~ bit 0

CTMAH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **CTMAH**: CTM CCRA High Byte Register bit 1 ~ bit 0
CTM 10-bit CCRA bit 9 ~ bit 8

CTMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	CTPAU	CTCK2	CTCK1	CTCK0	CTON	CTRP2	CTRP1	CTRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **CTPAU:** CTM Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **CTCK2~CTCK0:** Select CTM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: CTCK rising edge clock
111: CTCK falling edge clock
These three bits are used to select the clock source for the TM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3 **TnON:** CTM Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the CTOC bit, when the CTON bit changes from low to high.

Bit 2~0 **CTRP2~CTRP0:** CTM CCRP 3-bit register, compared with the CTM Counter bit 9~bit 7 Comparator P Match Period
000: 1024 CTM clocks
001: 128 CTM clocks
010: 256 CTM clocks
011: 384 CTM clocks
100: 512 CTM clocks
101: 640 CTM clocks
110: 768 CTM clocks
111: 896 CTM clocks
These three bits are used to setup the value on the internal CCRP 3-bit register, which are then compared with the internal counter's highest three bits. The result of this comparison can be selected to clear the internal counter if the CTCCLR bit is set to zero. Setting the CTCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest three counter bits, the compare values exist in 128 clock cycle multiples. Clearing all three bits to zero is in effect allowing the counter to overflow at its maximum value.

CTMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	CTM1	CTM0	CTIO1	CTIO0	CTOC	CTPOL	CTDPX	CTCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **CTM1~CTM0**: Select CTM Operating Mode
00: Compare Match Output Mode
01: Undefined
10: PWM Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the CTM1 and CTM0 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **CTIO1~CTIO0**: Select CTP output function

Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output

PWM Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Undefined

Timer/counter Mode
unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the CTIO1 and CTIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the CTOC bit in the CTMC1 register. Note that the output level requested by the CTIO1 and CTIO0 bits must be different from the initial value setup using the CTOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the CTON bit from low to high.

In the PWM Mode, the CTIO1 and CTIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the CTIO1 and CTIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the CTIO1 and CTIO0 bits are changed when the TM is running.

Bit 3	**CTOC**: CTP Output control bit Compare Match Output Mode 0: Initial low 1: Initial high PWM Mode 0: Active low 1: Active high This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.
Bit 2	**CTPOL**: CTP Output polarity Control 0: Non-invert 1: Invert This bit controls the polarity of the CTP output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.
Bit 1	**CTDPX**: CTM PWM duty/period Output Polarity Control 0: CCRP - period; CCRA - duty 1: CCRP - duty; CCRA - period This bit, determines which of the CCRA and CCRP registers are used for period and duty control of the PWM waveform.
Bit 0	**CTCCLR**: Select CTM Counter clear condition 0: CTM Comparatror P match 1: CTM Comparatror A match This bit is used to select the method which clears the counter. Remember that the Compact TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the CTCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The CTCCLR bit is not used in the PWM Mode.

Compact Type TM Operating Modes

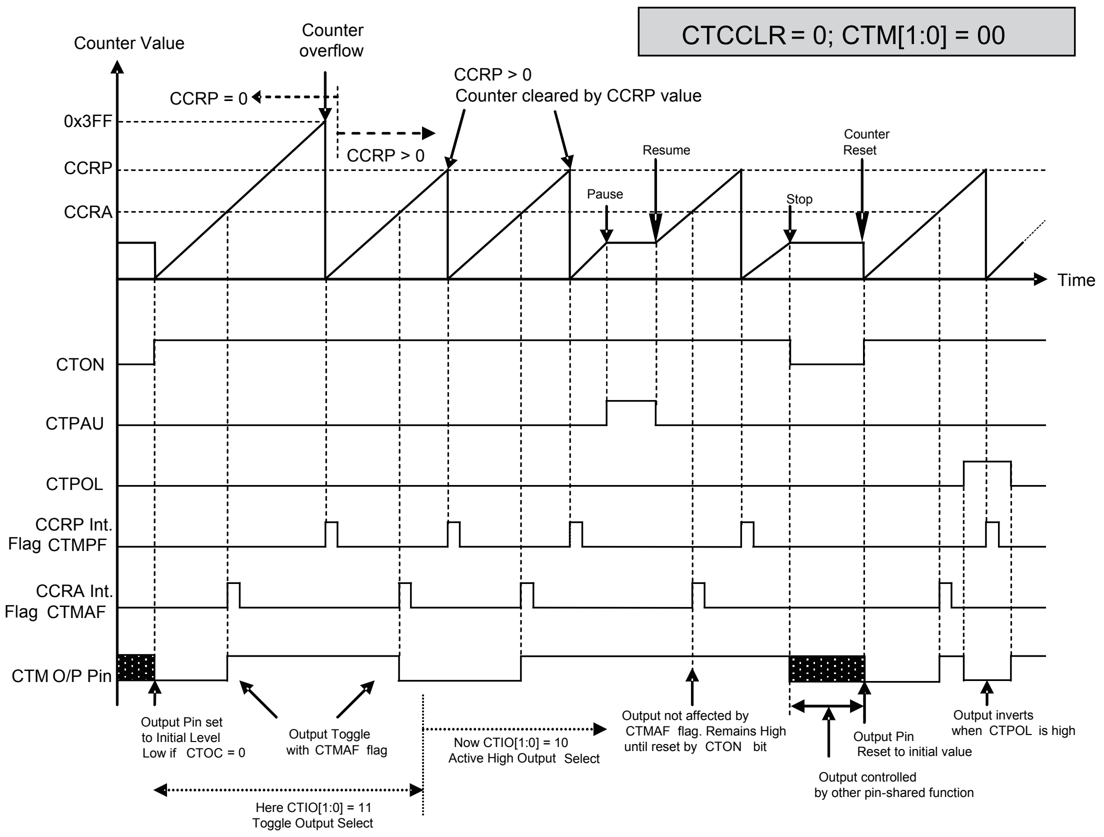
The Compact Type TM can operate in one of three operating modes, Compare Match Output Mode, PWM Mode or Timer/Counter Mode. The operating mode is selected using the CTM1 and CTM0 bits in the CTMC1 register.

Compare Match Output Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the CTCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both CTMAF and CTMPF interrupt request flags for the Comparator A and Comparator P respectively, will both be generated.

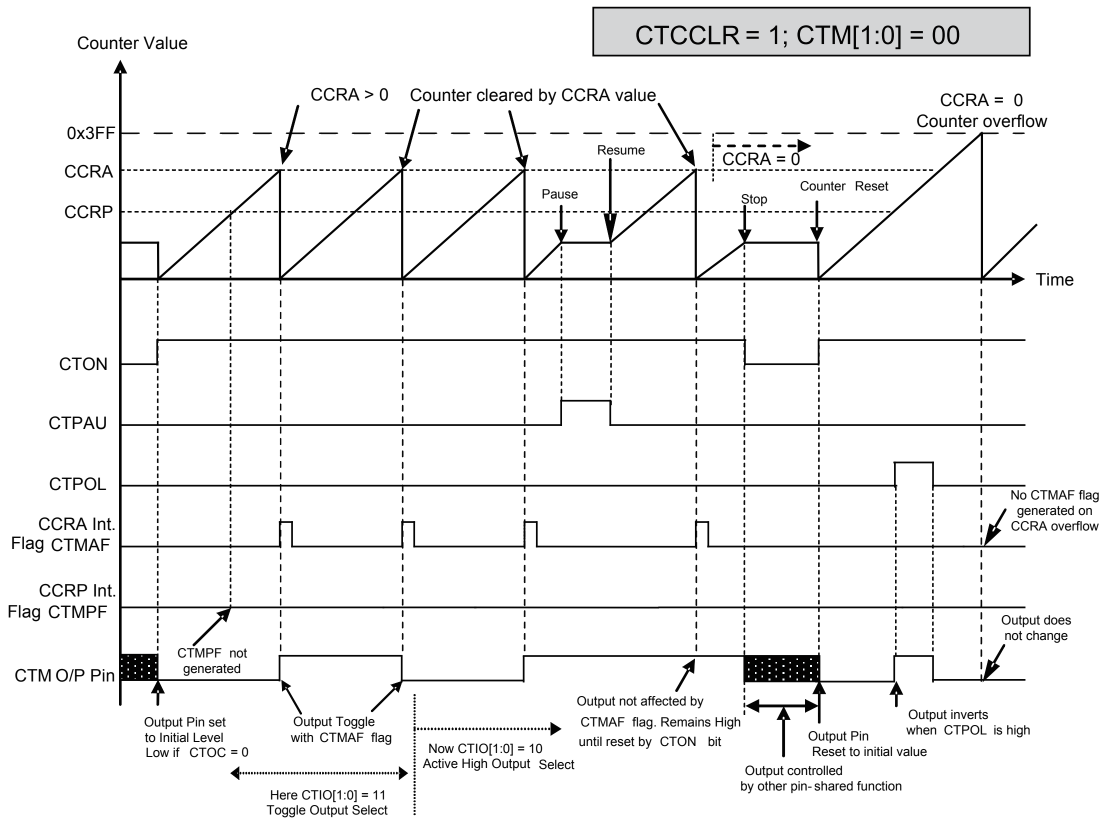
If the CTCCLR bit in the CTMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the CTMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when CTCCLR is high no CTMPF interrupt request flag will be generated. If the CCRA bits are all zero, the counter will overflow when its reaches its maximum 10-bit, 3FF Hex, value, however here the CTMAF interrupt request flag will not be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin will change state. The TM output pin condition however only changes state when a CTMAF interrupt request flag is generated after a compare match occurs from Comparator A. The CTMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the CTIO1 and CTIO0 bits in the CTMC1 register. The TM output pin can be selected using the CTIO1 and CTIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the TnON bit changes from low to high, is setup using the CTOC bit. Note that if the CTIO1 and CTIO0 bits are zero then no pin change will take place.


CTCCLR = 0; CTM[1:0] = 00
Counter Value
Counter overflow
CCRP = 0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0x3FF
CCRP
CCRA
Pause
Resume
Stop
Counter Reset
Time
CTON
CTPAU
CTPOL
CCRP Int. Flag CTMPF
CCRA Int. Flag CTMAF
CTM O/P Pin
Output Pin set to Initial Level Low if CTOC = 0
Output Toggle with CTMAF flag
Now CTIO[1:0] = 10 Active High Output Select
Output not affected by CTMAF flag. Remains High until reset by CTON bit
Output Pin Reset to initial value
Output inverts when CTPOL is high
Output controlled by other pin-shared function
Here CTIO[1:0] = 11 Toggle Output Select

Compare Match Output Mode – CTCCLR=0

Note: 1. With CTCCLR=0, a Comparator P match will clear the counter
2. The TM output pin is controlled only by the CTMAF flag
3. The output pin is reset to its initial state by a CTON bit rising edge


CTCCLR = 1; CTM[1:0] = 00
Counter Value
0x3FF
CCRA
CCRP
CCRA > 0
Counter cleared by CCRA value
CCRA = 0
Counter overflow
Resume
Pause
CCRA = 0
Counter Reset
Stop
Time
CTON
CTPAU
CTPOL
CCRA Int. Flag CTMAF
CCRP Int. Flag CTMPF
CTM O/P Pin
No CTMAF flag generated on CCRA overflow
Output does not change
CTMPF not generated
Output Pin set to Initial Level Low if CTOC = 0
Output Toggle with CTMAF flag
Here CTIO[1:0] = 11 Toggle Output Select
Now CTIO[1:0] = 10 Active High Output Select
Output not affected by CTMAF flag. Remains High until reset by CTON bit
Output controlled by other pin-shared function
Output Pin Reset to initial value
Output inverts when CTPOL is high

Compare Match Output Mode – CTCCLR=1

Note: 1. With **CTCCLR**=1, a Comparator A match will clear the counter
2. The TM output pin is controlled only by the CTMAF flag
3. The output pin is reset to its initial state by a CTON bit rising edge
4. The CTMPF flag is not generated when CTCCLR=1

Timer/Counter Mode

To select this mode, bits CTM1 and CTM0 in the CTMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To selec t this mode, bits CTM1 and CTM0 in the CTMC1 register should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the CTCCLR bit has no effect on the PWM operation. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. Which register is used to control either frequency or duty cycle is determined using the CTDPX bit in the CTMC1 register. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The CTOC bit in the CTMC1 register is used to select the required polarity of the PWM waveform while the two CTIO1 and CTIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The CTPOL bit is used to reverse the polarity of the PWM output waveform.

CTM, PWM Mode, Edge-aligned Mode, CTDPX=0

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	128	256	384	512	640	768	896	1024
Duty	CCRA							

If f_{SYS}=16MHz, TM clock source is $f_{SYS}/4$, CCRP=100b and CCRA=128,

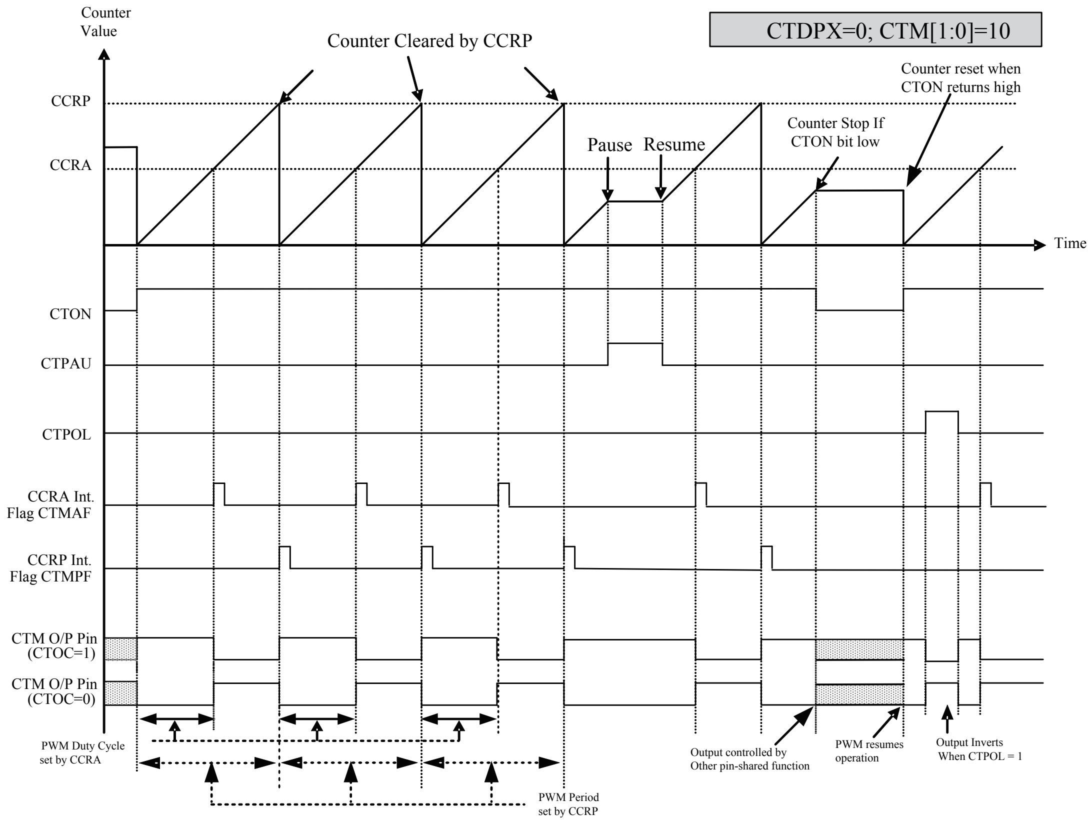
The CTM PWM output frequency=(f_{SYS}/4)/512=f_{SYS}/2048=7.8125kHz, duty=128/512=25%.

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.

CTM, PWM Mode, Edge-aligned Mode, CTDPX=1

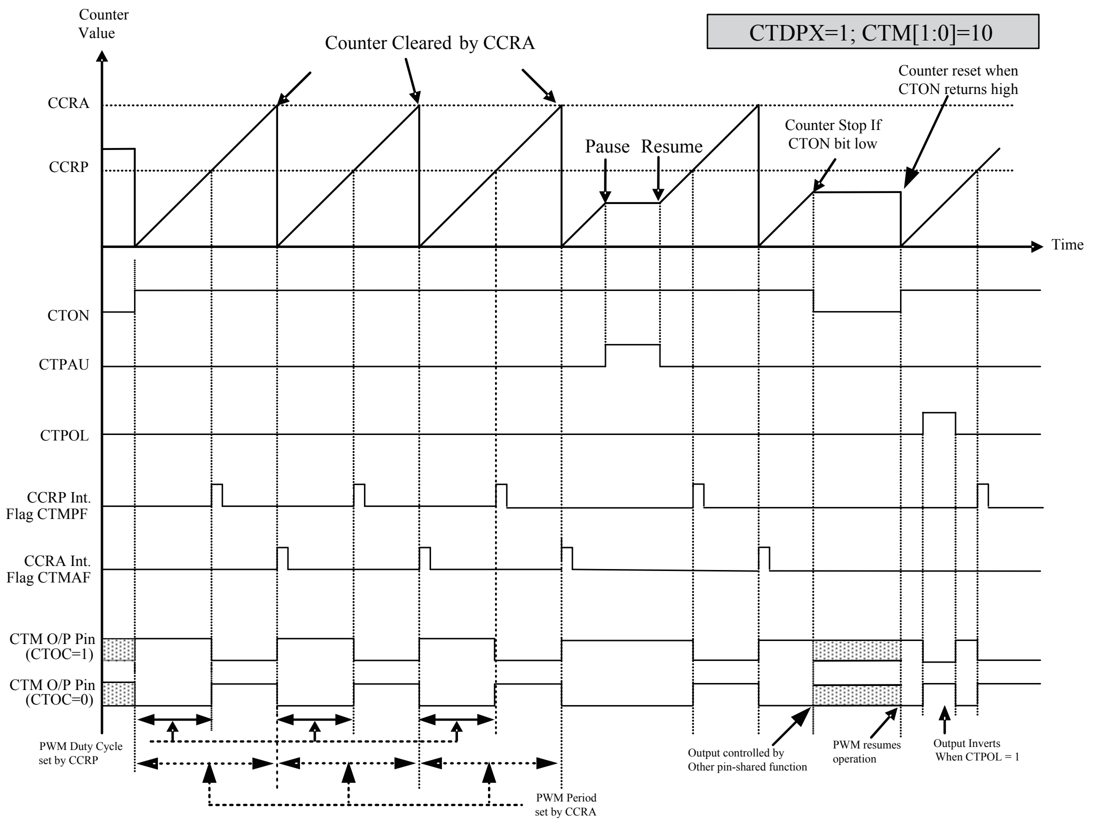
CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	CCRA							
Duty	128	256	384	512	640	768	896	1024

The PWM output period is determined by the CCRA register value together with the TM clock while the PWM duty cycle is defined by the CCRP register value.



PWM Mode – CTDPX=0

Note: 1. Here CTDPX=0 -- Counter cleared by CCRP
2. A counter clear sets the PWM Period
3. The internal PWM function continues running even when CTIO [1:0]=00 or 01
4. The CTCCLR bit has no influence on PWM operation



PWM Mode – CTDPX=1

Note: 1. Here CTDPX =1 – Counter cleared by CCRA
2. A counter clear sets the PWM Period
3. The internal PWM function continues running even when CTIO [1:0]=00 or 01
4. The CTCCLR bit has no influence on PWM operation

Periodic Type TM – PTM

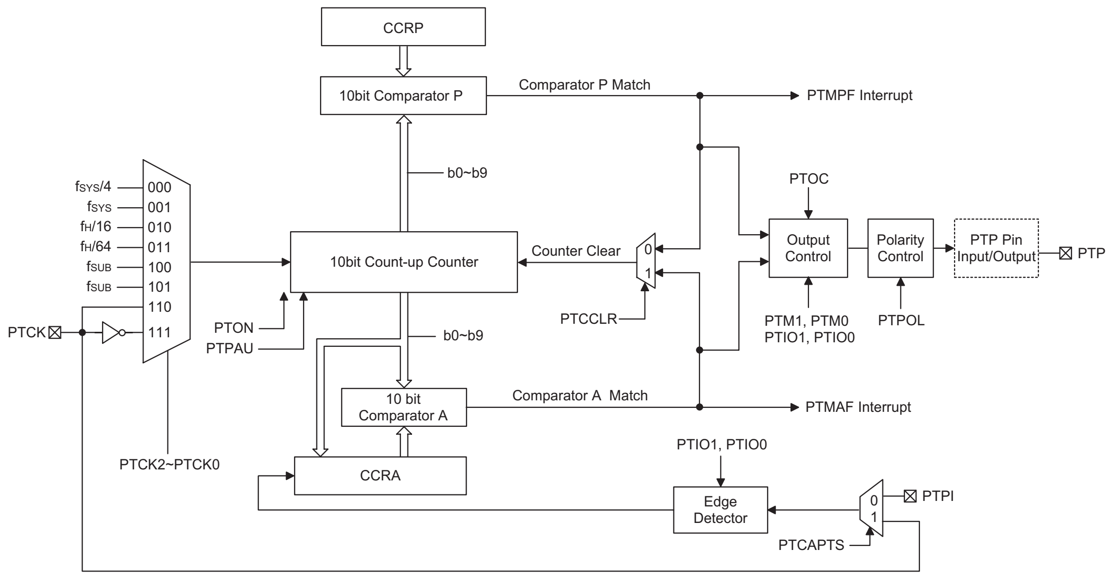
The Periodic Type TM contains five operating modes, which are Compare Match Output, Timer/ Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Periodic TM can also be controlled with one or two external input pins and can drive one external output pin.

TM Type	TM Input Pin	TM Output Pin
10-bit PTM	PTCK, PTPI	PTP

Periodic TM Operation

At the core is a 10-bit count-up counter which is driven by a user selectable internal or external clock source. There are also two internal comparators with the names, Comparator A and Comparator P. These comparators will compare the value in the counter with CCRP and CCRA registers. The CCRP or CCRA comparator is 10-bit wide whose value is compared with all counter bits.

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the PTON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Periodic Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control an output pin. All operating setup conditions are selected using relevant internal registers.



Periodic Type TM Block Diagram

Periodic Type TM Register Description

Overall operation of the Periodic TM is controlled using a series of registers. A read only register pair exists to store the internal counter 10-bit value, while two read/write register pairs exist to store the internal 10-bit CCRA and CCRP value. The remaining two registers are control registers which setup the different operating and control modes.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PTMC0	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
PTMC1	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
PTMDL	D7	D6	D5	D4	D3	D2	D1	D0
PTMDH	—	—	—	—	—	—	D9	D8
PTMAL	D7	D6	D5	D4	D3	D2	D1	D0
PTMAH	—	—	—	—	—	—	D9	D8
PTMRPL	D7	D6	D5	D4	D3	D2	D1	D0
PTMRPH	—	—	—	—	—	—	D9	D8

Periodic TM Register List

PTMDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **PTMDL:** PTM Counter Low Byte Register bit 7~bit 0
PTM 10-bit Counter bit 7~bit 0

PTMDH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **PTMDH:** PTM Counter High Byte Register bit 1~bit 0
PTM 10-bit Counter bit 9~bit 8

PTMAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **PTMAL:** PTM CCRA Low Byte Register bit 7~bit 0
PTM 10-bit CCRA bit 7~bit 0

PTMAH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **PTMAH:** PTM CCRA High Byte Register bit 1~bit 0
PTM 10-bit CCRA bit 9~bit 8

PTMRPL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **PTMRPL:** PTM CCRP Low Byte Registger bit 7 ~ bit 0
PTM 10-bit CCRP bit 7 ~ bit 0

PTMRPH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"
Bit 1~0 **PTMRPH:** PTM CCRP High Byte Registger bit 1 ~ bit 0
PTM 10-bit CCRP bit 9 ~ bit 8

PTMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	PTPAU	PTCK2	PTCK1	PTCK0	PTON	—	—	—
R/W	R/W	R/W	R/W	R/W	R/W	—	—	—
POR	0	0	0	0	0	—	—	—

Bit 7 **PTPAU:** PTM Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **PTCK2~PTCK0:** Select PTM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: PTCK rising edge clock
111: PTCK falling edge clock
These three bits are used to select the clock source for the TM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can

be found in the oscillator section.

Bit 3 **PTON:** PTM Counter On/Off Control
0: Off
1: On
This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again. If the TM is in the Compare Match Output Mode then the TM output pin will be reset to its initial condition, as specified by the T2OC bit, when the PTOC bit changes from low to high.

Bit 2~0 Unimplemented, read as "0"

PTMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	PTM1	PTM0	PTIO1	PTIO0	PTOC	PTPOL	PTCAPTS	PTCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **PTM1~PTM0:** Select PTM Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode
These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the PTM1 and PTM2 bits. In the Timer/Counter Mode, the TM output pin control must be disabled.

Bit 5~4 **PTIO1~PTIO0:** Select PTP output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of PTPI or PTCK
01: Input capture at falling edge of PTPI or PTCK
10: Input capture at falling/rising edge of PTPI or PTCK
11: Input capture disabled
Timer/counter Mode
Unused
These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.
In the Compare Match Output Mode, the PTIO1 and PTIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the PTOC bit in the PTMC1 register. Note that the output level requested by the PTIO1 and PTIO0 bits must be different from the initial value setup using the PTOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its

initial level by changing the level of the PTON bit from low to high.

In the PWM Mode, the PTIO1 and PTIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the PTIO1 and PTIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the PTIO1 and PTIO0 bits are changed when the TM is running.

Bit 3 **PTOC:** PTP Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high

This is the output control bit for the TM output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **PTPOL:** PTP Output polarity Control
0: Non-invert
1: Invert

This bit controls the polarity of the PTP output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1 **PTCAPTS:** PTM Capture trigger source select
0: From PTPI pin
1: From PTCK pin

Bit 0 **PTCCLR:** Select PTM Counter clear condition
0: PTM Comparator P match
1: PTM Comparator A match

This bit is used to select the method which clears the counter. Remember that the Periodic TM contains two comparators, Comparator A and Comparator P, either of which can be selected to clear the internal counter. With the PTCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The PTCCLR bit is not used in the PWM, Single Pulse or Input Capture Mode.

Periodic Type TM Operating Modes

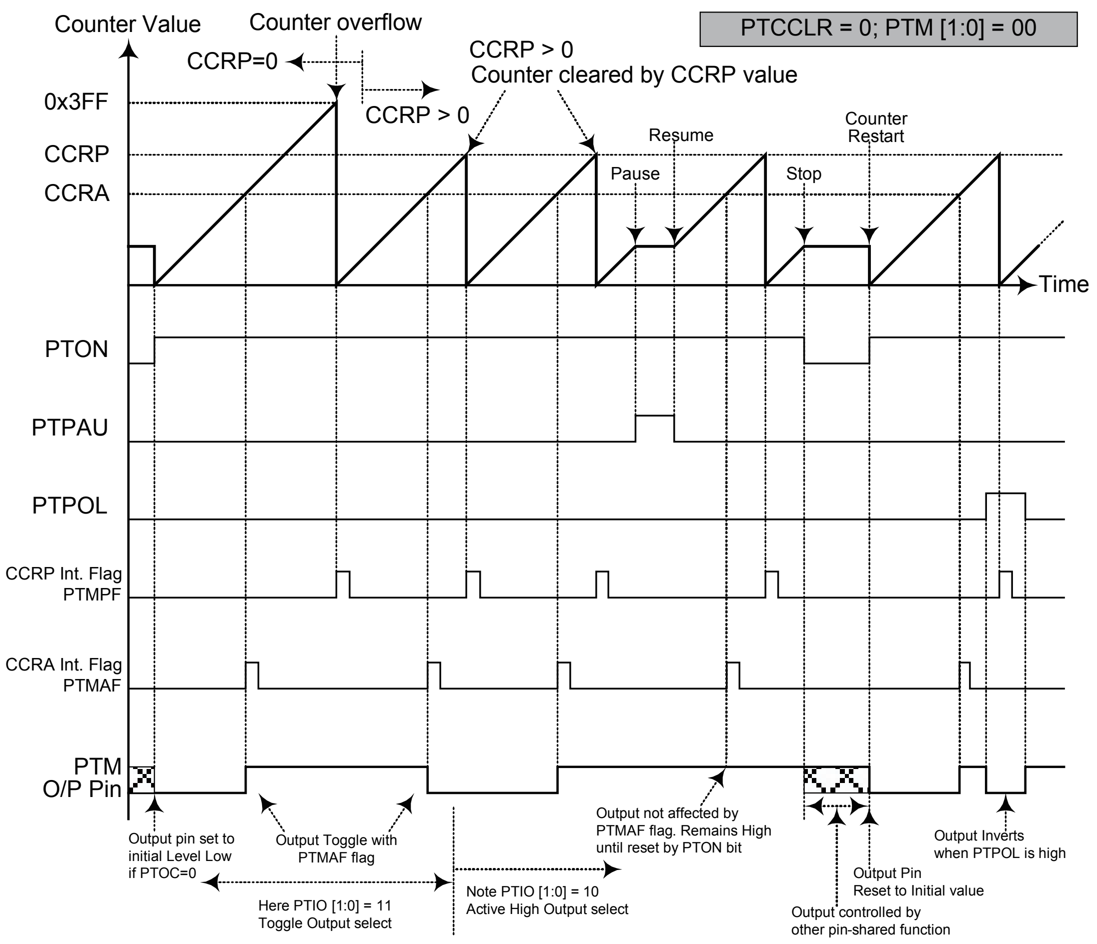
The Periodic Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the PTM1 and PTM0 bits in the PTMC1 register.

Compare Output Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register, should be set to 00 respectively. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the PTCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both PTMAF and PTMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

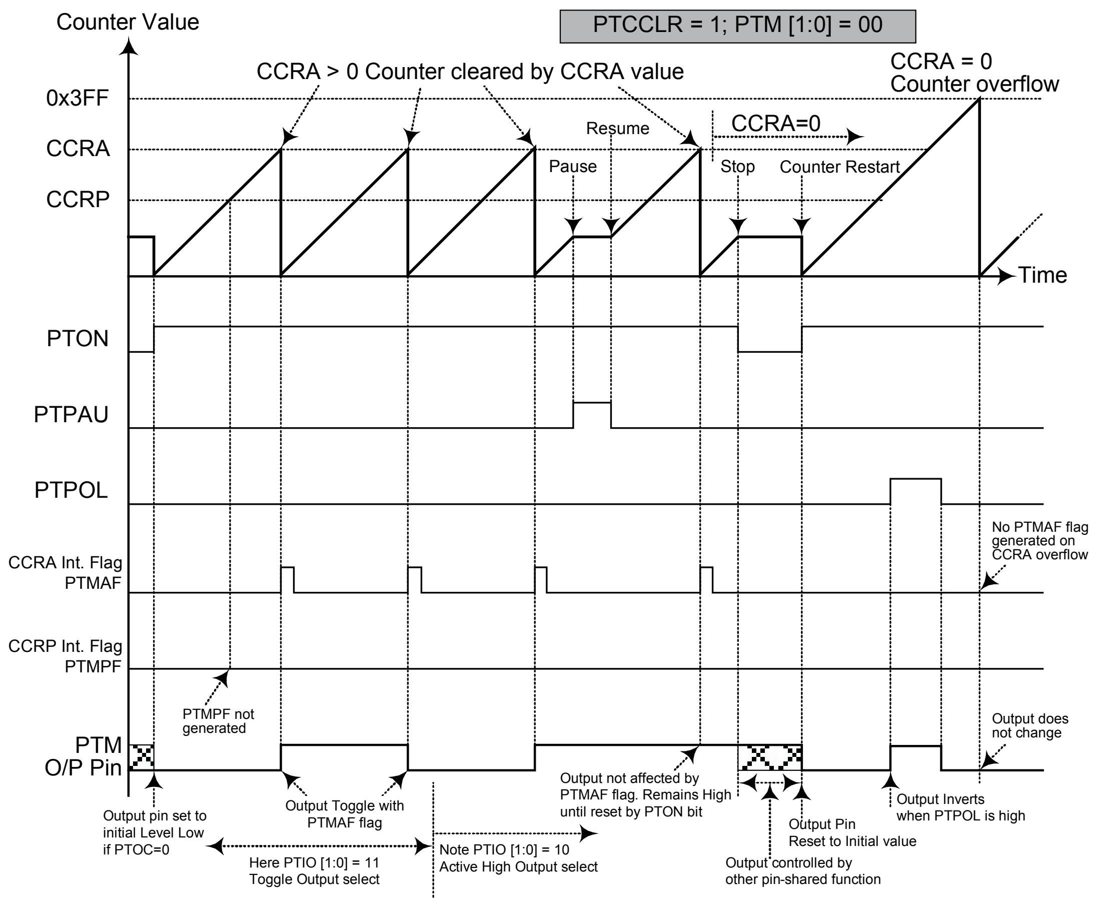
If the PTCCLR bit in the PTMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the PTMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when PTCCLR is high no PTMPF interrupt request flag will be generated. In the Compare Match Output Mode, the CCRA can not be set to "0".

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a PTMAF interrupt request flag is generated after a compare match occurs from Comparator A. The PTMPF interrupt request flag, generated from a compare match occurs from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state are determined by the condition of the PTIO1 and PTIO0 bits in the PTMC1 register. The TM output pin can be selected using the PTIO1 and PTIO0 bits to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A. The initial condition of the TM output pin, which is setup after the PTON bit changes from low to high, is setup using the PTOC bit. Note that if the PTIO1 and PTIO0 bits are zero then no pin change will take place.


Counter Value
Counter overflow
PTCCLR = 0; PTM [1:0] = 00
CCRP=0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0x3FF
CCRP
CCRA
Resume
Pause
Counter Restart
Stop
Time
PTON
PTPAU
PTPOL
CCRP Int. Flag PTMPF
CCRA Int. Flag PTMAF
PTM O/P Pin
Output pin set to initial Level Low if PTOC=0
Output Toggle with PTMAF flag
Here PTIO [1:0] = 11 Toggle Output select
Note PTIO [1:0] = 10 Active High Output select
Output not affected by PTMAF flag. Remains High until reset by PTON bit
Output controlled by other pin-shared function
Output Pin Reset to Initial value
Output Inverts when PTPOL is high

Compare Match Output Mode – PTCCLR = 0

Note: 1. With PTCCLR=0, a Comparator P match will clear the counter
2. The TM output pin is controlled only by the PTMAF flag
3. The output pin is reset to its initial state by a PTON bit rising edge


Counter Value
PTCCLR = 1; PTM [1:0] = 00
CCRA > 0 Counter cleared by CCRA value
CCRA = 0
Counter overflow
0x3FF
CCRA
CCRP
Resume
CCRA=0
Pause
Stop
Counter Restart
Time
PTON
PTPAU
PTPOL
CCRA Int. Flag
PTMAF
No PTMAF flag generated on CCRA overflow
CCRP Int. Flag
PTMPF
PTMPF not generated
PTM O/P Pin
Output does not change
Output not affected by PTMAF flag. Remains High until reset by PTON bit
Output Toggle with PTMAF flag
Output pin set to initial Level Low if PTOC=0
Output Pin Reset to Initial value
Output Inverts when PTPOL is high
Here PTIO [1:0] = 11 Toggle Output select
Note PTIO [1:0] = 10 Active High Output select
Output controlled by other pin-shared function

Compare Match Output Mode – PTCCLR = 1

Note: 1. With PTCCLR=1, a Comparator A match will clear the counter
2. The TM output pin is controlled only by the PTMAF flag
3. The output pin is reset to its initial state by a PTON bit rising edge
4. A PTMPF flag is not generated when PTCCLR =1

Timer/Counter Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 11 respectively. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared function.

PWM Output Mode

To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively and also the PTIO1 and PTIO0 bits should be set to 10 respectively. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the PTCCLR bit has no effect as the PWM period. Both of the CCRA and CCRP registers are used to generate the PWM waveform, one register is used to clear the internal counter and thus control the PWM waveform frequency, while the other one is used to control the duty cycle. The PWM waveform frequency and duty cycle can therefore be controlled by the values in the CCRA and CCRP registers.

An interrupt flag, one for each of the CCRA and CCRP, will be generated when a compare match occurs from either Comparator A or Comparator P. The PTOC bit in the PTMC1 register is used to select the required polarity of the PWM waveform while the two PTIO1 and PTIO0 bits are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The PTPOL bit is used to reverse the polarity of the PWM output waveform.

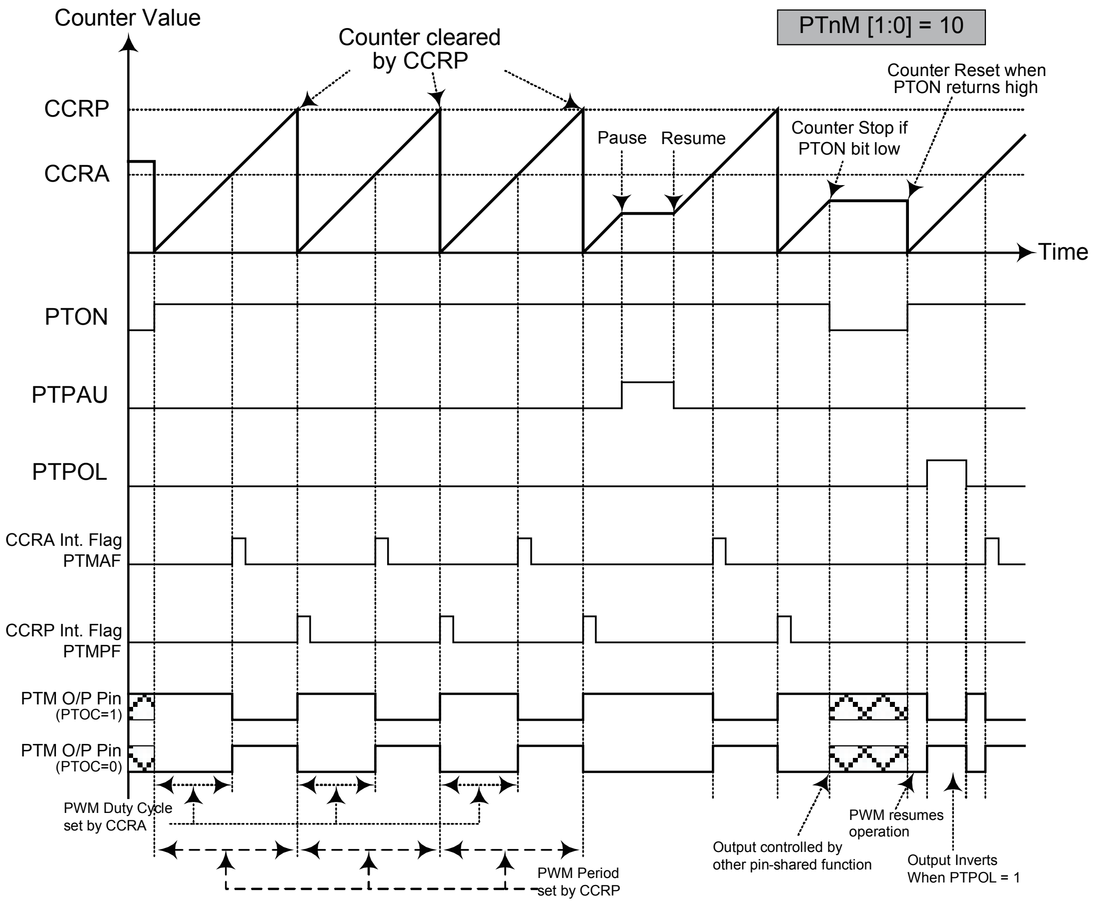
PTM, PWM Mode, Edge-aligned Mode

Period		Duty
CCRP=0 → Period = 1024	CCRP=1~1023 → Period = 1~1023	CCRA

If f_{SYS}=16MHz, TM clock source select f_{SYS}/4, CCRP=512 and CCRA=128,

The PTM PWM output frequency=(f_{SYS}/4)/512=f_{SYS}/2048=7.8125kHz, duty=128/512=25%

If the Duty value defined by the CCRA register is equal to or greater than the Period value, then the PWM output duty is 100%.



PWM Mode

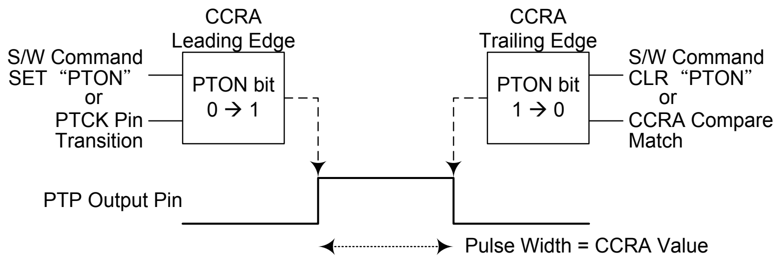
Note: 1. The counter is cleared by CCRP.
2. A counter clear sets the PWM Period
3. The internal PWM function continues running even when PTIO [1:0] = 00 or 01
4. The PTCCLR bit has no influence on PWM operation

Single Pulse Output Mode

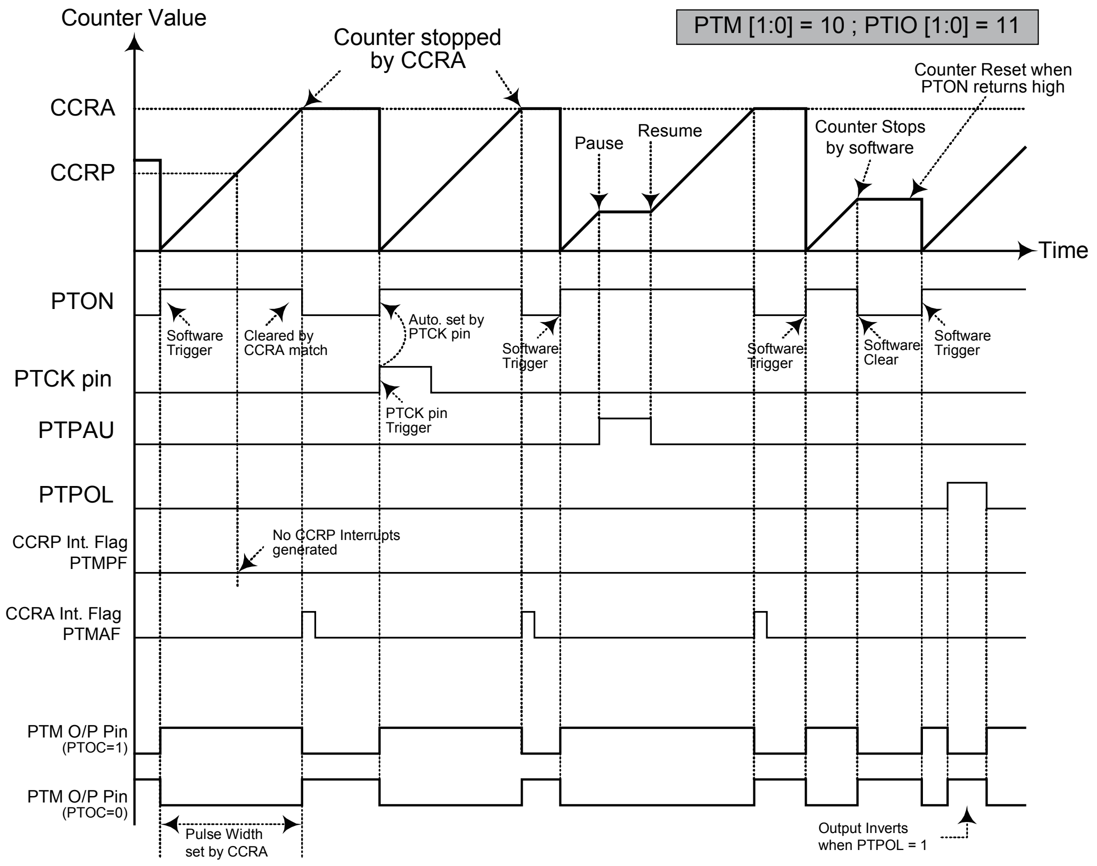
To select this mode, bits PTM1 and PTM0 in the PTMC1 register should be set to 10 respectively and also the PTIO1 and PTIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse output leading edge is a low to high transition of the PTON bit, which can be implemented using the application program. However in the Single Pulse Mode, the PTON bit can also be made to automatically change from low to high using the external PTCK pin, which will in turn initiate the Single Pulse output. When the PTON bit transitions to a high level, the counter will start running and the pulse leading edge will be generated. The PTON bit should remain high when the pulse is in its active state. The generated pulse trailing edge will be generated when the PTON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.

However a compare match from Comparator A will also automatically clear the PTON bit and thus generate the Single Pulse output trailing edge. In this way the CCRA value can be used to control the pulse width. A compare match from Comparator A will also generate a TM interrupt. The counter can only be reset back to zero when the PTON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The PTCCLR bit is not used in this Mode.



Single Pulse Generation



Single Pulse Mode

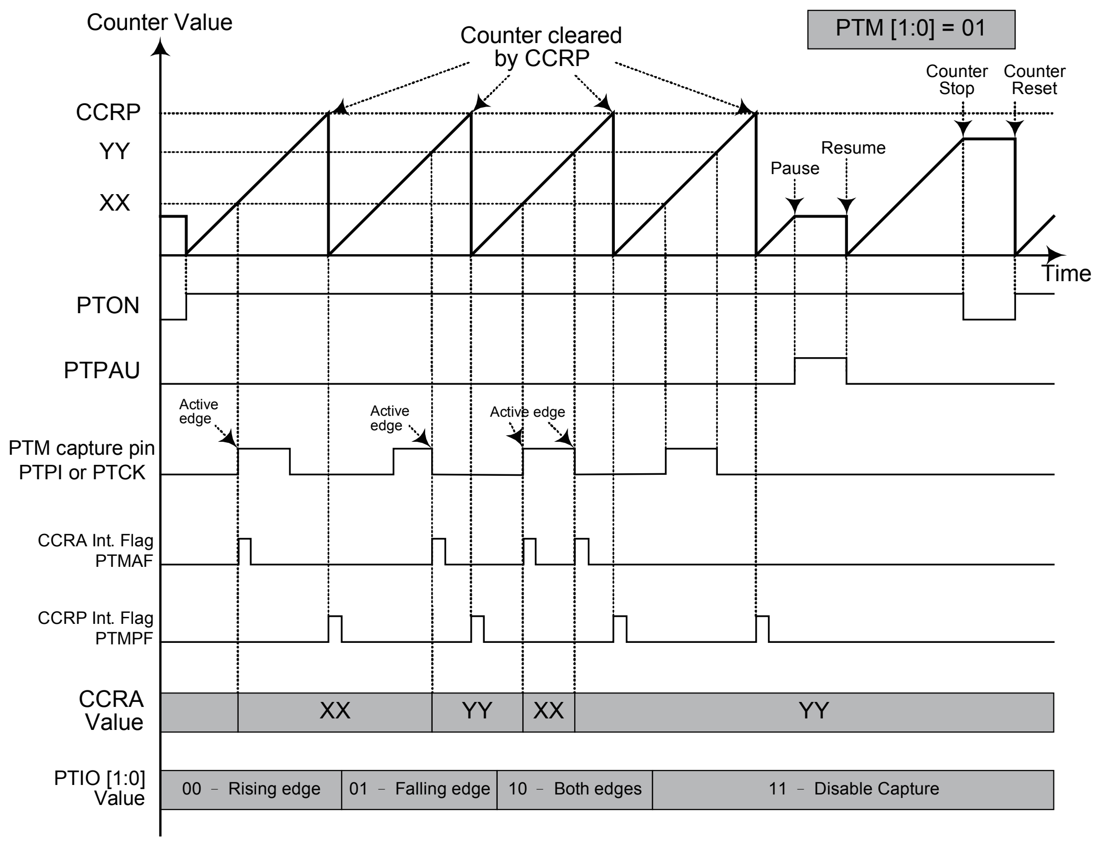
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse triggered by the PTCK pin or by setting the PTON bit high
4. A PTCK pin active edge will automatically set the PTON bit high.
5. In the Single Pulse Mode, PTIO [1:0] must be set to "11" and can not be changed.

Capture Input Mode

To select this mode bits PTM1 and PTM0 in the PTMC1 register should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the PTPI or PTCK pin, selected by the PTCAPTS bit in the PTMC0 register. The capture input pin active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the PTIO1 and PTIO0 bits in the PTMC1 register. The counter is started when the PTON bit changes from low to high which is initiated using the application program.

When the required edge transition appears on the PTPI or PTCK pin the present value in the counter will be latched into the CCRA registers and a TM interrupt generated. Irrespective of what events occur on the PTPI or PTCK pin the counter will continue to free run until the PTON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The PTIO1 and PTIO0 bits can select the active trigger edge on the PTPI or PTCK pin to be a rising edge, falling edge or both edge types. If the PTIO1 and PTIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the PTPI or PTCK pin, however it must be noted that the counter will continue to run.

As the PTPI or PTCK pin is pin shared with other functions, care must be taken if the TM is in the Capture Input Mode. This is because if the pin is setup as an output, then any transitions on this pin may cause an input capture operation to be executed. The PTCCLR, PTOC and PTPOL bits are not used in this Mode.



Capture Input Mode

Note: 1. PTM [1:0] = 01 and active edge set by the PTIO [1:0] bits

2. A TM Capture input pin active edge transfers the counter value to CCRA

3. PTCCLR bit not used

4. No output function – PTOC and PTPOL bits are not used

5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero.

Enhanced Type TM – ETM

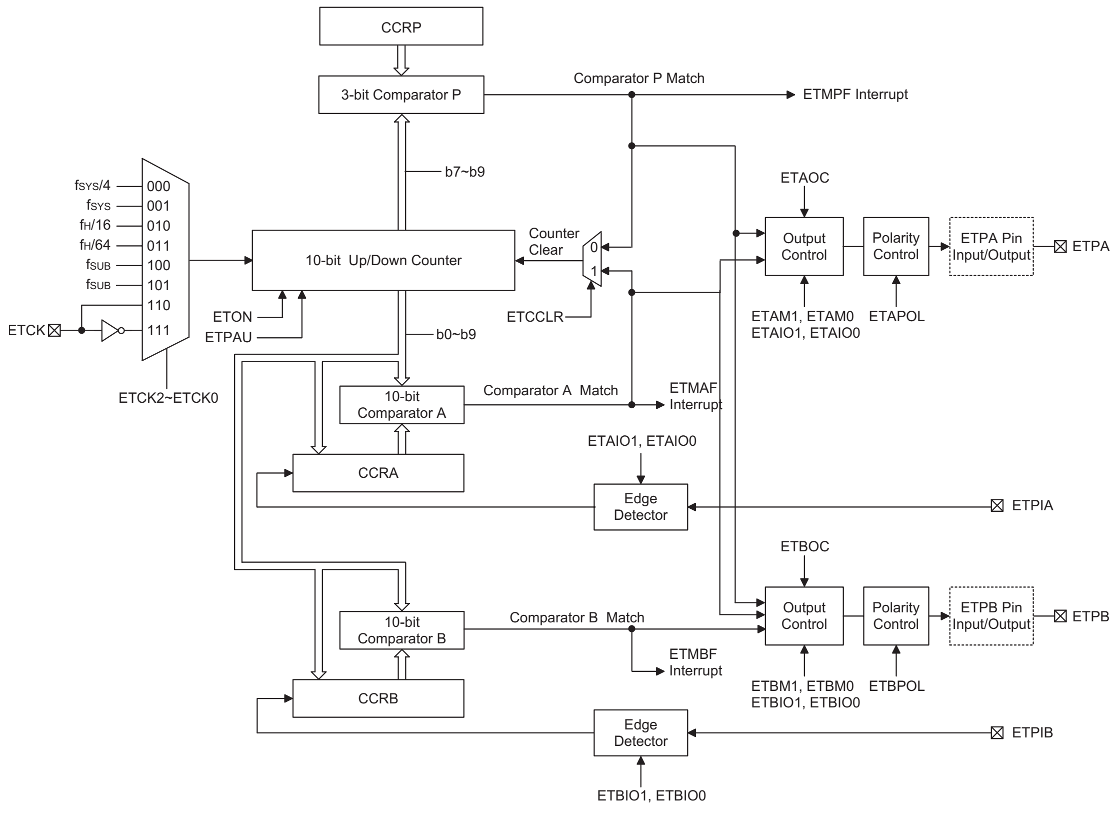
The Enhanced Type TM contains five operating modes, which are Compare Match Output, Timer/Event Counter, Capture Input, Single Pulse Output and PWM Output modes. The Enhanced TM can also be controlled with an external input pin and can drive two external output pins.

TM Type	TM Input Pin	TM Output Pin
10-bit ETM	ETCK, ETPIA. ETPIB	ETPA, ETPB

Enhanced TM Operation

At its core is a 10-bit count-up/count-down counter which is driven by a user selectable internal or external clock source. There are three internal comparators with the names, Comparator A, Comparator B and Comparator P. These comparators will compare the value in the counter with the CCRA, CCRB and CCRP registers. The CCRP comparator is 3-bits wide whose value is compared with the highest 3-bits in the counter while CCRA and CCRB are 10-bits wide and therefore compared with all counter bits.

The only way of changing the value of the 10-bit counter using the application program, is to clear the counter by changing the ETON bit from low to high. The counter will also be cleared automatically by a counter overflow or a compare match with one of its associated comparators. When these conditions occur, a TM interrupt signal will also usually be generated. The Enhanced Type TM can operate in a number of different operational modes, can be driven by different clock sources including an input pin and can also control output pins. All operating setup conditions are selected using relevant internal registers.



Enhanced Type TM Block Diagram

Enhanced Type TM Register Description

Overall operation of the Enhanced TM is controlled using a series of registers. A read only register pair exists to store the internal counter 10-bit value, while two read/write register pairs exist to store the internal 10-bit CCRA and CCRB value. The remaining three registers are control registers which setup the different operating and control modes as well as the three CCRP bits.

Name	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
ETMC0	ETPAU	ETCK2	ETCK1	ETCK0	ETON	ETRP2	ETRP1	ETRP0
ETMC1	ETAM1	ETAM0	ETAIO1	ETAIO0	ETAOC	ETAPOL	ETCDN	ETCCLR
ETMC2	ETBM1	ETBM0	ETBIO1	ETBIO0	ETBOC	ETBPOL	ETPWM1	ETPWM0
ETMDL	D7	D6	D5	D4	D3	D2	D1	D0
ETMDH	—	—	—	—	—	—	D9	D8
ETMAL	D7	D6	D5	D4	D3	D2	D1	D0
ETMAH	—	—	—	—	—	—	D9	D8
ETMBL	D7	D6	D5	D4	D3	D2	D1	D0
ETMBH	—	—	—	—	—	—	D9	D8

Enhanced TM Register List

ETMC0 Register

Bit	7	6	5	4	3	2	1	0
Name	ETPAU	ETCK2	ETCK1	ETCK0	ETON	ETRP2	ETRP1	ETRP0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **ETPAU:** ETM Counter Pause Control
0: Run
1: Pause
The counter can be paused by setting this bit high. Clearing the bit to zero restores normal counter operation. When in a Pause condition the TM will remain powered up and continue to consume power. The counter will retain its residual value when this bit changes from low to high and resume counting from this value when the bit changes to a low value again.

Bit 6~4 **ETCK2~ETCK0:** Select ETM Counter clock
000: $f_{SYS}/4$
001: f_{SYS}
010: $f_H/16$
011: $f_H/64$
100: f_{SUB}
101: f_{SUB}
110: ETCK rising edge clock
111: ETCK falling edge clock
These three bits are used to select the clock source for the TM. The external pin clock source can be chosen to be active on the rising or falling edge. The clock source f_{SYS} is the system clock, while f_H and f_{SUB} are other internal clocks, the details of which can be found in the oscillator section.

Bit 3	**ETON:** ETM Counter On/Off Control 0: Off 1: On This bit controls the overall on/off function of the TM. Setting the bit high enables the counter to run, clearing the bit disables the TM. Clearing this bit to zero will stop the counter from counting and turn off the TM which will reduce its power consumption. When the bit changes state from low to high the internal counter value will be reset to zero, however when the bit changes from high to low, the internal counter will retain its residual value until the bit returns high again. If the TM is in the Compare Match Output Mode then the TM output pins will be reset to initial conditions, as specified by the ETAOC or ETBOC bit, when the ETON bit changes from low to high.
Bit 2~0	**ETRP2~ETRP0:** ETM CCRP 3-bit register, compared with the ETM Counter bit 9~bit 7 Comparator P Match Period 000: 1024 ETM clocks 001: 128 ETM clocks 010: 256 ETM clocks 011: 384 ETM clocks 100: 512 ETM clocks 101: 640 ETM clocks 110: 768 ETM clocks 111: 896 ETM clocks These three bits are used to setup the value on the internal CCRP 3-bit register, which are then compared with the internal counter highest three bits. The result of this comparison can be selected to clear the internal counter if the ETCCLR bit is set to zero. Setting the ETCCLR bit to zero ensures that a compare match with the CCRP values will reset the internal counter. As the CCRP bits are only compared with the highest three counter bits, the compare values exist in 128 clock cycle multiples. Clearing all three bits to zero is in effect allowing the counter to overflow at its maximum value.

ETMC1 Register

Bit	7	6	5	4	3	2	1	0
Name	ETAM1	ETAM0	ETAIO1	ETAIO0	ETAOC	ETAPOL	ETCDN	ETCCLR
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **ETAM1~ETAM0:** Select ETM CCRA Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter Mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the ETAM1 and ETAM0 bits. In the Timer/Counter Mode, the TM output pin is not used.

Bit 5~4 **ETAIO1~ETAIO0:** Select ETPA or ETPIA pin output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM output inactive state
01: PWM output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of ETPIA
01: Input capture at falling edge of ETPIA
10: Input capture at falling/rising edge of ETPIA
11: Input capture disabled
Timer/counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the ETAIO1 and ETAIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the ETAOC bit in the ETMC1 register. Note that the output level requested by the ETAIO1 and ETAIO0 bits must be different from the initial value setup using the ETAOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the ETON bit from low to high.

In the PWM Mode, the ETAIO1 and ETAIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the ETAIO1 and ETAIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the ETAIO1 and ETAIO0 bits are changed when the TM is running.

Bit 3	**ETAOC:** ETPA Output control bit Compare Match Output Mode 0: Initial low 1: Initial high PWM Mode/Single Pulse Output Mode 0: Active low 1: Active high This is the output control bit for the ETPA output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.
Bit 2	**ETAPOL:** ETPA Output polarity Control 0: Non-invert 1: Invert This bit controls the polarity of the ETPA output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.
Bit 1	**ETCDN:** ETM Counter count up or down flag 0: Count up 1: Count down
Bit 0	**ETCCLR:** Select ETM Counter clear condition 0: ETM Comparator P match 1: ETM Comparator A match This bit is used to select the method which clears the counter. Remember that the Enhanced TM contains three comparators, Comparator A, Comparator B and Comparator P, but only Comparator A or Comparator P can be selected to clear the internal counter. With the ETCCLR bit set high, the counter will be cleared when a compare match occurs from the Comparator A. When the bit is low, the counter will be cleared when a compare match occurs from the Comparator P or with a counter overflow. A counter overflow clearing method can only be implemented if the CCRP bits are all cleared to zero. The ETCCLR bit is not used in the Single Pulse Output or Capture Input Mode.

ETMC2 Register

Bit	7	6	5	4	3	2	1	0
Name	ETBM1	ETBM0	ETBIO1	ETBIO0	ETBOC	ETBPOL	ETPWM1	ETPWM0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~6 **ETBM1~ETBM0:** Select ETM CCRB Operating Mode
00: Compare Match Output Mode
01: Capture Input Mode
10: PWM Mode or Single Pulse Output Mode
11: Timer/Counter mode

These bits setup the required operating mode for the TM. To ensure reliable operation the TM should be switched off before any changes are made to the ETBM1 and ETBM0 bits. In the Timer/Counter Mode, the TM output pin control must is not used.

Bit 5~4 **ETBIO1~ETBIO0:** Select ETPB or ETPIB pin output function
Compare Match Output Mode
00: No change
01: Output low
10: Output high
11: Toggle output
PWM Mode/Single Pulse Output Mode
00: PWM Output inactive state
01: PWM Output active state
10: PWM output
11: Single pulse output
Capture Input Mode
00: Input capture at rising edge of ETPIB
01: Input capture at falling edge of ETPIB
10: Input capture at falling/rising edge of ETPIB
11: Input capture disabled
Timer/counter Mode
Unused

These two bits are used to determine how the TM output pin changes state when a certain condition is reached. The function that these bits select depends upon in which mode the TM is running.

In the Compare Match Output Mode, the ETBIO1 and ETBIO0 bits determine how the TM output pin changes state when a compare match occurs from the Comparator A. The TM output pin can be setup to switch high, switch low or to toggle its present state when a compare match occurs from the Comparator A. When the bits are both zero, then no change will take place on the output. The initial value of the TM output pin should be setup using the ETBOC bit in the ETMC2 register. Note that the output level requested by the ETBIO1 and ETBIO0 bits must be different from the initial value setup using the ETBOC bit otherwise no change will occur on the TM output pin when a compare match occurs. After the TM output pin changes state, it can be reset to its initial level by changing the level of the ETON bit from low to high.

In the PWM Mode, the ETBIO1 and ETBIO0 bits determine how the TM output pin changes state when a certain compare match condition occurs. The PWM output function is modified by changing these two bits. It is necessary to only change the values of the ETBIO1 and ETBIO0 bits only after the TMn has been switched off. Unpredictable PWM outputs will occur if the ETBIO1 and ETBIO0 bits are changed when the TM is running.

Bit 3 **ETBOC:** ETPB Output control bit
Compare Match Output Mode
0: Initial low
1: Initial high
PWM Mode/Single Pulse Output Mode
0: Active low
1: Active high
This is the output control bit for the ETPB output pin. Its operation depends upon whether TM is being used in the Compare Match Output Mode or in the PWM Mode/ Single Pulse Output Mode. It has no effect if the TM is in the Timer/Counter Mode. In the Compare Match Output Mode it determines the logic level of the TM output pin before a compare match occurs. In the PWM Mode it determines if the PWM signal is active high or active low.

Bit 2 **ETBPOL:** ETPB Output polarity Control
0: Non-invert
1: Invert
This bit controls the polarity of the ETPB output pin. When the bit is set high the TM output pin will be inverted and not inverted when the bit is zero. It has no effect if the TM is in the Timer/Counter Mode.

Bit 1~0 **ETPWM1~ETPWM0:** Select ETM PWM mode
00: Edge aligned
01: Centre aligned, compare match on count up
10: Centre aligned, compare match on count down
11: Centre aligned, compare match on count up or down

ETMDL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	0	0	0	0

Bit 7~0 **ETMDL:** ETM Counter Low Byte Register bit 7~bit 0
ETM 10-bit Counter bit 7~bit 0

ETMDH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R	R
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **ETMDH:** ETM Counter High Byte Register bit 1~bit 0
ETM 10-bit Counter bit 9~bit 8

ETMAL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **ETMAL:** ETM CCRA Low Byte Register bit 7~bit 0
ETM 10-bit CCRA bit 7~bit 0

ETMAH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **ETMAH:** ETM CCRA High Byte Register bit 1~bit 0
ETM 10-bit CCRA bit 9~bit 8

ETMBL Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~0 **ETMBL:** ETM CCRB Low Byte Register bit 7~bit 0
ETM 10-bit CCRB bit 7~bit 0

ETMBH Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	—	—	D9	D8
R/W	—	—	—	—	—	—	R/W	R/W
POR	—	—	—	—	—	—	0	0

Bit 7~2 Unimplemented, read as "0"

Bit 1~0 **ETMBH:** ETM CCRB High Byte Register bit 1~bit 0
ETM 10-bit CCRB bit 9~bit 8

Enhanced Type TM Operating Modes

The Enhanced Type TM can operate in one of five operating modes, Compare Match Output Mode, PWM Output Mode, Single Pulse Output Mode, Capture Input Mode or Timer/Counter Mode. The operating mode is selected using the ETAM1 and ETAM0 bits in the ETMC1, and the ETBM1 and ETBM0 bits in the ETMC2 register.

ETM Operating Mode	CCRA Compare Match Output Mode	CCRA Timer/ Counter Mode	CCRA PWM Output Mode	CCRA Single Pulse Output Mode	CCRA Input Capture Mode
CCRB Compare Match Output Mode	√	—	—	—	—
CCRB Timer/Counter Mode	—	√	—	—	—
CCRB PWM Output Mode	—	—	√	—	—
CCRB Single Pulse Output Mode	—	—	—	√	—
CCRB Input Capture Mode	—	—	—	—	√

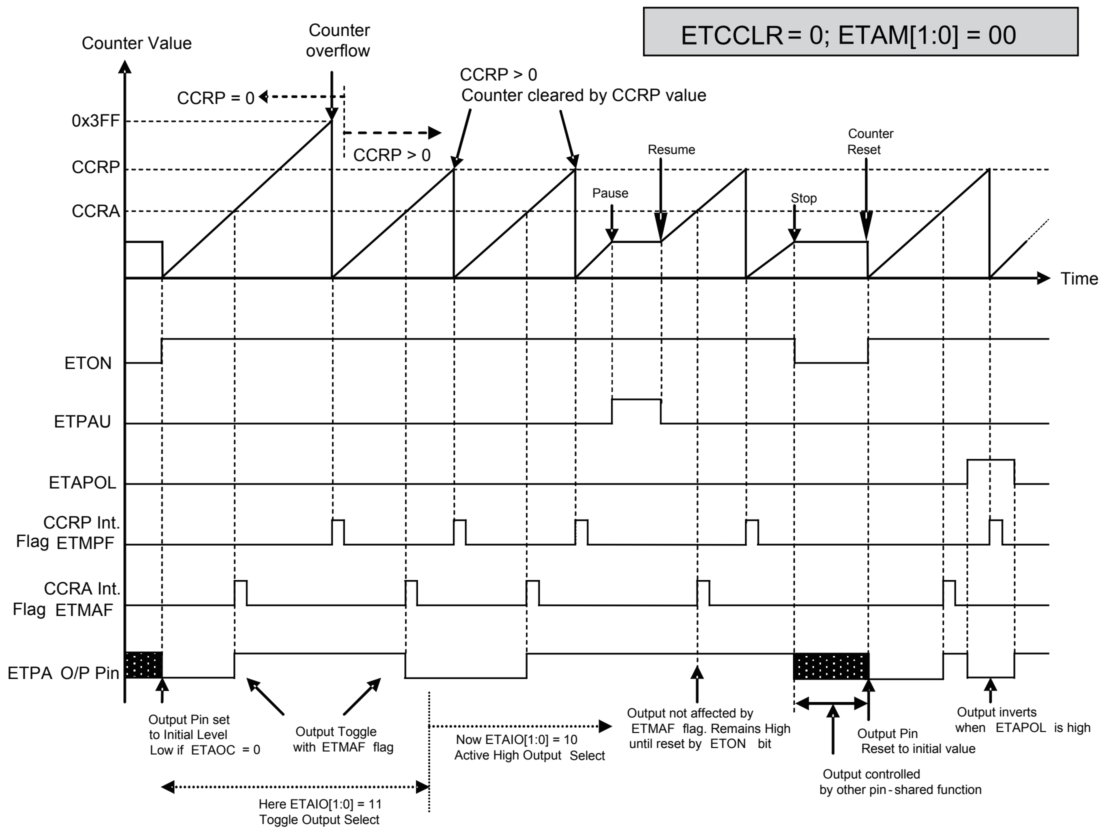
"√": permitted; "—": not permitt

Compare Output Mode

To select this mode, bits ETAM1, ETAM0 and ETBM1, ETBM0 in the ETMC1/ETMC2 registers should be all cleared to zero. In this mode once the counter is enabled and running it can be cleared by three methods. These are a counter overflow, a compare match from Comparator A and a compare match from Comparator P. When the ETCCLR bit is low, there are two ways in which the counter can be cleared. One is when a compare match occurs from Comparator P, the other is when the CCRP bits are all zero which allows the counter to overflow. Here both the ETMAF and ETMPF interrupt request flags for Comparator A and Comparator P respectively, will both be generated.

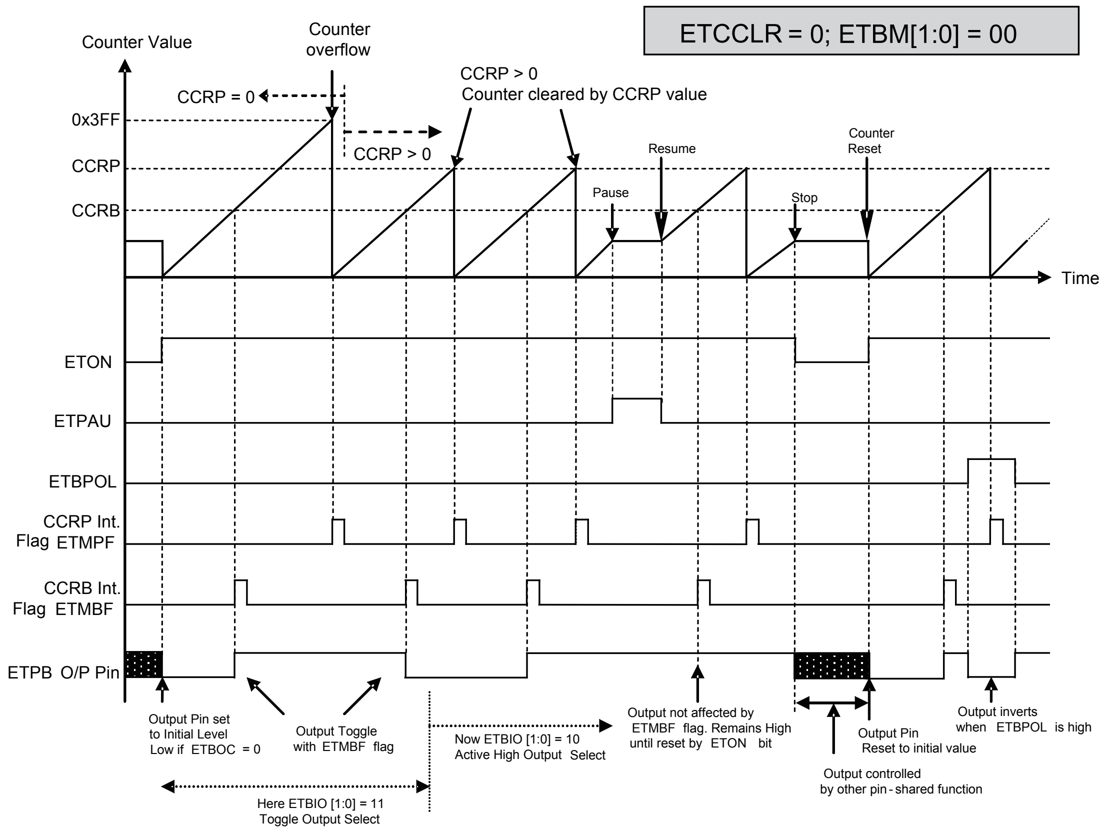
If the ETCCLR bit in the ETMC1 register is high then the counter will be cleared when a compare match occurs from Comparator A. However, here only the ETMAF interrupt request flag will be generated even if the value of the CCRP bits is less than that of the CCRA registers. Therefore when ETCCLR is high no ETMPF interrupt request flag will be generated.

As the name of the mode suggests, after a comparison is made, the TM output pin, will change state. The TM output pin condition however only changes state when a ETMAF or ETMBF interrupt request flag is generated after a compare match occurs from Comparator A or Comparator B. The ETMPF interrupt request flag, generated from a compare match from Comparator P, will have no effect on the TM output pin. The way in which the TM output pin changes state is determined by the condition of the ETAIO1 and ETAIO0 bits in the ETMC1 register for ETM CCRA, and the ETBIO1 and ETBIO0 bits in the ETMC2 register for ETM CCRB. The TM output pin can be selected using the ETAIO1, ETAIO0 bits (for the ETPA pin) and ETBIO1, ETBIO0 bits (for the ETPB pin) to go high, to go low or to toggle from its present condition when a compare match occurs from Comparator A or a compare match occurs from Comparator B. The initial condition of the TM output pin, which is setup after the ETON bit changes from low to high, is setup using the ETAOC or ETBOC bit for ETPA or ETPB output pin. Note that if the ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits are zero then no pin change will take place.


ETCCLR = 0; ETAM[1:0] = 00
Counter Value
Counter overflow
CCRP = 0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0x3FF
CCRP
CCRA
Pause
Resume
Stop
Counter Reset
Time
ETON
ETPAU
ETAPOL
CCRP Int. Flag ETMPF
CCRA Int. Flag ETMAF
ETPA O/P Pin
Output Pin set to Initial Level Low if ETAOC = 0
Output Toggle with ETMAF flag
Now ETAIO[1:0] = 10 Active High Output Select
Output not affected by ETMAF flag. Remains High until reset by ETON bit
Output Pin Reset to initial value
Output controlled by other pin-shared function
Output inverts when ETAPOL is high
Here ETAIO[1:0] = 11 Toggle Output Select

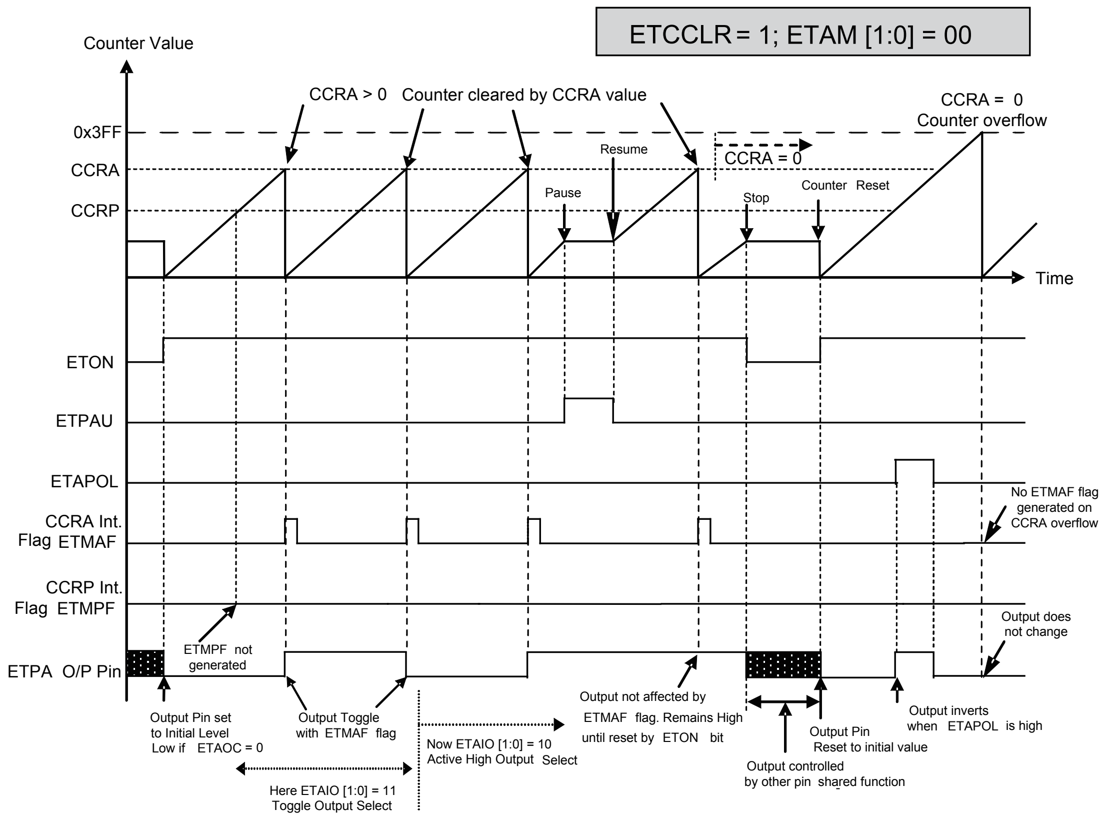
ETM CCRA Compare Match Output Mode – ETCCLR=0

Note: 1. With ETCCLR=0, a Comparator P match will clear the counter
2. The ETPA output pin is controlled only by the ETMAF flag
3. The output pin is reset to its initial state by a ETON bit rising edge


ETCCLR = 0; ETBM[1:0] = 00
Counter Value
Counter overflow
CCRP = 0
CCRP > 0
CCRP > 0
Counter cleared by CCRP value
0x3FF
CCRP
CCRB
Pause
Resume
Stop
Counter Reset
Time
ETON
ETPAU
ETBPOL
CCRP Int. Flag ETMPF
CCRB Int. Flag ETMBF
ETPB O/P Pin
Output Pin set to Initial Level Low if ETBOC = 0
Output Toggle with ETMBF flag
Now ETBIO [1:0] = 10 Active High Output Select
Output not affected by ETMBF flag. Remains High until reset by ETON bit
Output Pin Reset to initial value
Output controlled by other pin-shared function
Output inverts when ETBPOL is high
Here ETBIO [1:0] = 11 Toggle Output Select

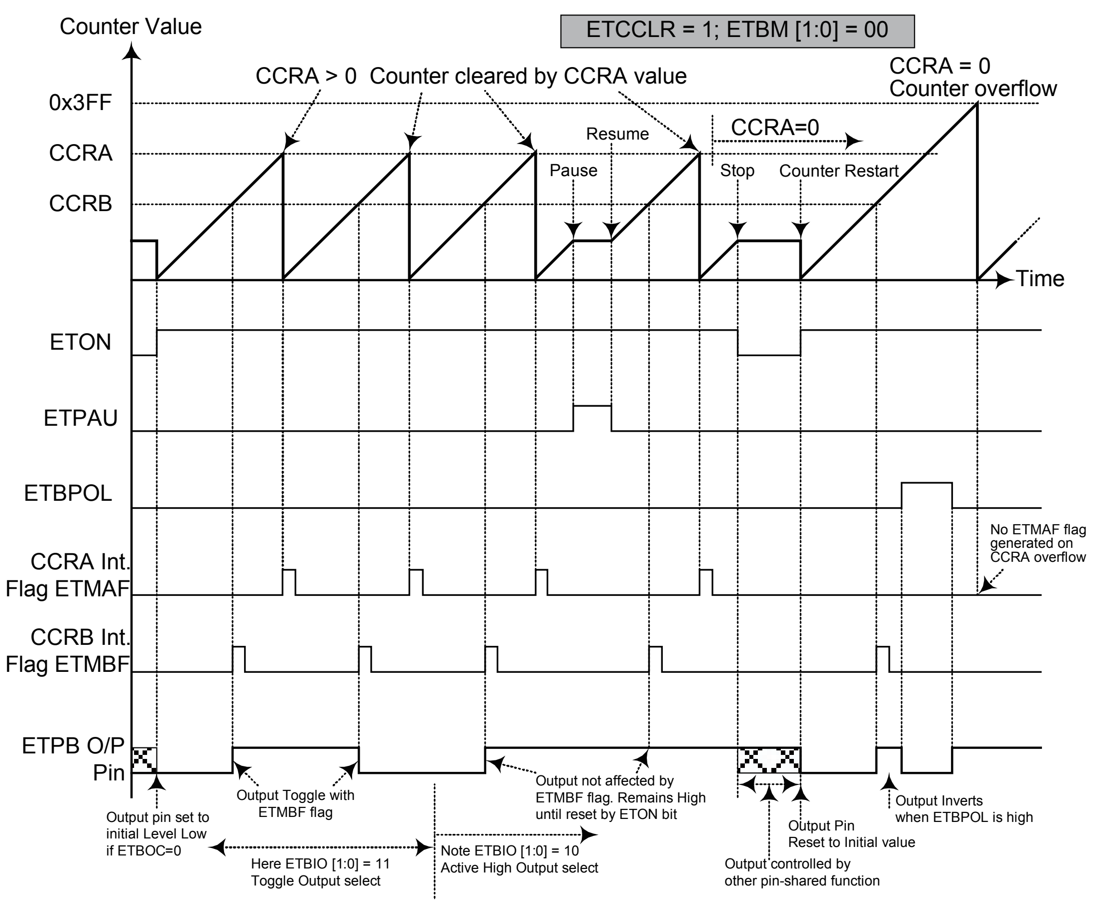
ETM CCRB Compare Match Output Mode – ETCCLR=0

Note: 1. With ETCCLR=0, a Comparator P match will clear the counter
2. The ETPB output pin is controlled only by the ETMBF flag
3. The output pin is reset to its initial state by a ETON bit rising edge


ETCCLR = 1; ETAM [1:0] = 00
Counter Value
CCRA > 0
Counter cleared by CCRA value
CCRA = 0
Counter overflow
0x3FF
CCRA
CCRP
Resume
Pause
CCRA = 0
Stop
Counter Reset
Time
ETON
ETPAU
ETAPOL
CCRA Int. Flag ETMAF
No ETMAF flag generated on CCRA overflow
CCRP Int. Flag ETMPF
Output does not change
ETPA O/P Pin
ETMPF not generated
Output Pin set to Initial Level Low if ETAOC = 0
Output Toggle with ETMAF flag
Here ETAIO [1:0] = 11 Toggle Output Select
Now ETAIO [1:0] = 10 Active High Output Select
Output not affected by ETMAF flag. Remains High until reset by ETON bit
Output controlled by other pin shared function
Output Pin Reset to initial value
Output inverts when ETAPOL is high

ETM CCRA Compare Match Output Mode – ETCCLR=1

Note: 1. With ETCCLR=1, a Comparator A match will clear the counter
2. The ETPA output pin is controlled only by the ETMAF flag
3. The ETPA output pin is reset to its initial state by a ETON bit rising edge
4. The ETMPF flag is not generated when ETCCLR=1


Counter Value
ETCCLR = 1; ETBM [1:0] = 00
CCRA > 0 Counter cleared by CCRA value
CCRA = 0
Counter overflow
0x3FF
CCRA
CCRB
Resume
CCRA=0
Pause
Stop
Counter Restart
Time
ETON
ETPAU
ETBPOL
CCRA Int. Flag ETMAF
No ETMAF flag generated on CCRA overflow
CCRB Int. Flag ETMBF
ETPB O/P Pin
Output pin set to initial Level Low if ETBOC=0
Output Toggle with ETMBF flag
Here ETBIO [1:0] = 11 Toggle Output select
Note ETBIO [1:0] = 10 Active High Output select
Output not affected by ETMBF flag. Remains High until reset by ETON bit
Output Pin Reset to Initial value
Output controlled by other pin-shared function
Output Inverts when ETBPOL is high

ETM CCRB Compare Match Output Mode – ETCCLR=1

Note: 1. With ETCCLR=1, a Comparator A match will clear the counter
2. The ETPB output pin is controlled only by the ETMBF flag
3. The ETPB output pin is reset to its initial state by a ETON bit rising edge
4. The ETMPF flag is not generated when ETCCLR=1

Timer/Counter Mode

To select this mode, bits ETAM1, ETAM0 and ETBM1, ETBM0 in the ETMC1 and ETMC2 registers should all be set high. The Timer/Counter Mode operates in an identical way to the Compare Match Output Mode generating the same interrupt flags. The exception is that in the Timer/Counter Mode the TM output pin is not used. Therefore the above description and Timing Diagrams for the Compare Match Output Mode can be used to understand its function. As the TM output pin is not used in this mode, the pin can be used as a normal I/O pin or other pin-shared functions.

PWM Output Mode

To select this mode, the required bit pairs, ETAM1, ETAM0 and ETBM1, ETBM0 in the ETMC1 and ETMC2 registers should be set to 10 respectively. If the ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits are also set to 10 respectively, the PWM waveform will be output on the TM output pin. The PWM function within the TM is useful for applications which require functions such as motor control, heating control, illumination control etc. By providing a signal of fixed frequency but of varying duty cycle on the TM output pin, a square wave AC waveform can be generated with varying equivalent DC RMS values.

As both the period and duty cycle of the PWM waveform can be controlled, the choice of generated waveform is extremely flexible. In the PWM mode, the ETCCLR bit is used to determine in which way the PWM period is controlled. With the ETCCLR bit set high, the PWM period can be finely controlled using the CCRA registers. In this case the CCRB registers are used to set the PWM duty value (for ETPB output pin). The CCRP bits are not used and ETPA output pin is not used. The PWM output can only be generated on the ETPB output pin. With the ETCCLR bit cleared to zero, the PWM period is set using one of the eight values of the three CCRP bits, in multiples of 128. Now both CCRA and CCRB registers can be used to setup different duty cycle values to provide dual PWM outputs on their relative ETPA and ETPB pins.

The ETPWM1 and ETPWM0 bits determine the PWM alignment type, which can be either edge or centre type. In edge alignment, the leading edge of the PWM signals will all be generated concurrently when the counter is reset to zero. With all power currents switching on at the same time, this may give rise to problems in higher power applications. In centre alignment the centre of the PWM active signals will occur sequentially, thus reducing the level of simultaneous power switching currents.

Interrupt flags, one for each of the CCRA, CCRB and CCRP, will be generated when a compare match occurs from the Comparator A, Comparator B or Comparator P. The ETAOC and ETBOC bits in the ETMC1 and ETMC2 register are used to select the required polarity of the PWM waveform while the two ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits pairs are used to enable the PWM output or to force the TM output pin to a fixed high or low level. The ETAPOL and ETBPOL bit are used to reverse the polarity of the PWM output waveform.

ETM, PWM Mode, Edge-aligned Mode, ETCCLR=0

CCRP	001b	010b	011b	100b	101b	110b	111b	000b
Period	128	256	384	512	640	768	896	1024
A Duty	CCRA							
B Duty	CCRB							

If f_{SYS}=12MHz, TM clock source select $f_{SYS}/4$, CCRP=100b, CCRA=128 and CCRB=256,

The ETPA PWM output frequency=$(f_{SYS}/4)/512=f_{SYS}/2048$=5.8594kHz, duty=128/512=25%.

The ETPB PWM output frequency=$(f_{SYS}/4)/512=f_{SYS}/2048$=5.8594kHz, duty=256/512=50%.

If the Duty value defined by CCRA or CCRB register is equal to or greater than the Period value, then the PWM output duty is 100%.

ETM, PWM Mode, Edge-aligned Mode, ETCCLR=1

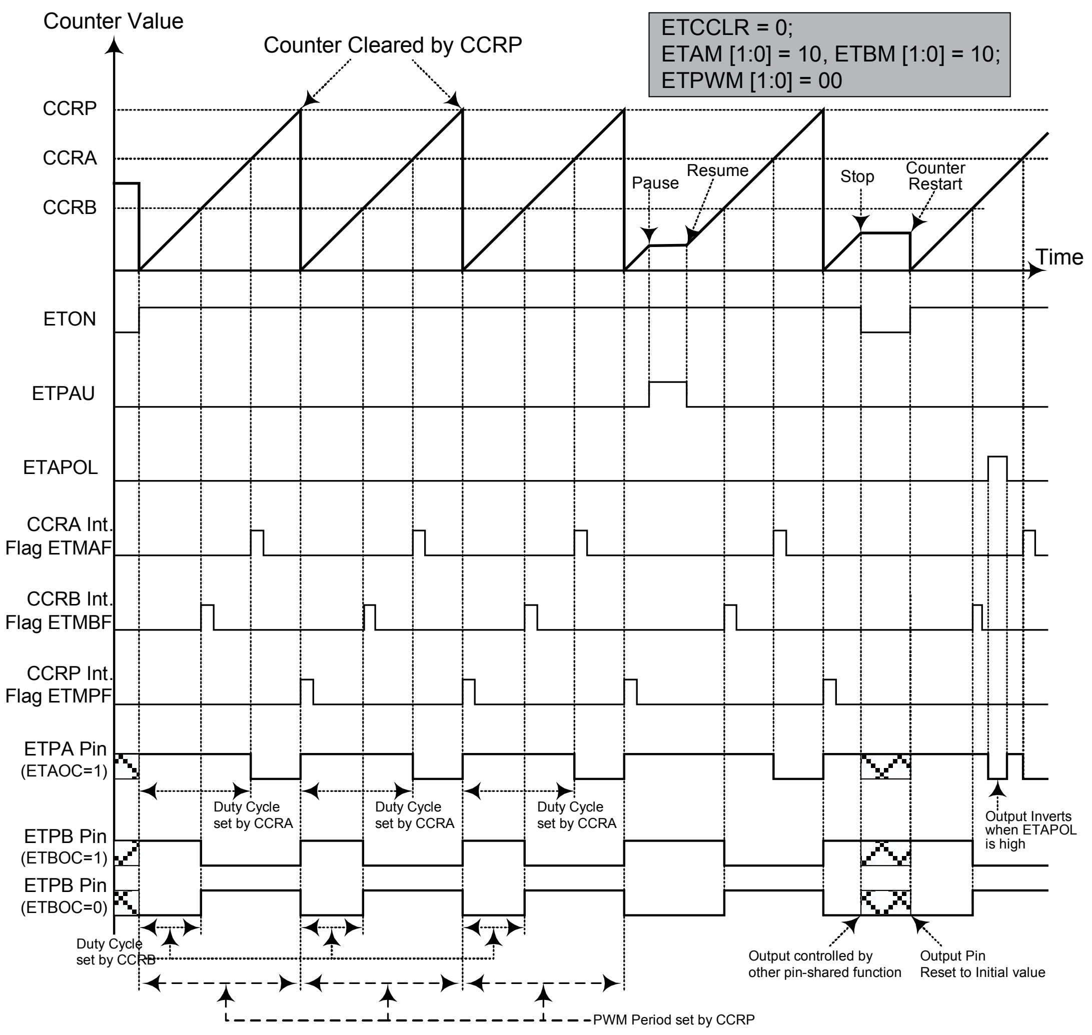
CCRA	1	2	3	511	512	1021	1022	1023
Period	1	2	3	511	512	1021	1022	1023
B Duty	CCRB							

ETM, PWM Mode, Center-aligned Mode, ETCCLR=0

CCRA	001b	010b	011b	100b	101b	110b	111b	000b
Period	256	512	768	1024	1280	1536	1792	2046
A Duty	(CCRA×2)-1							
B Duty	(CCRB×2)-1							

ETM, PWM Mode, Center-aligned Mode, ETCCLR=1

CCRA	1	2	3	511	512	1021	1022	1023
Period	2	4	6	1022	1024	2042	2044	2046
B Duty	(CCRB×2)-1							

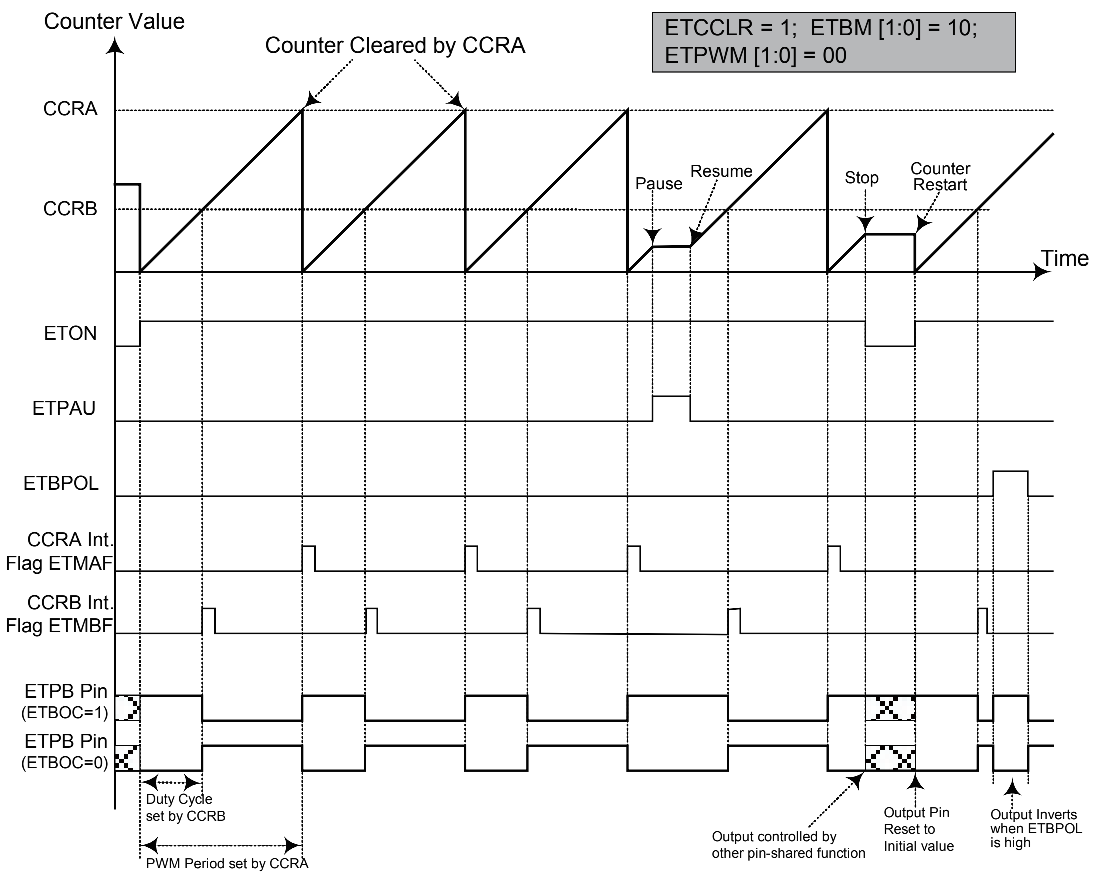


ETM PWM Mode – Edge Aligned

Note: 1. Here ETCCLR=0 therefore CCRP clears the counter and determines the PWM period

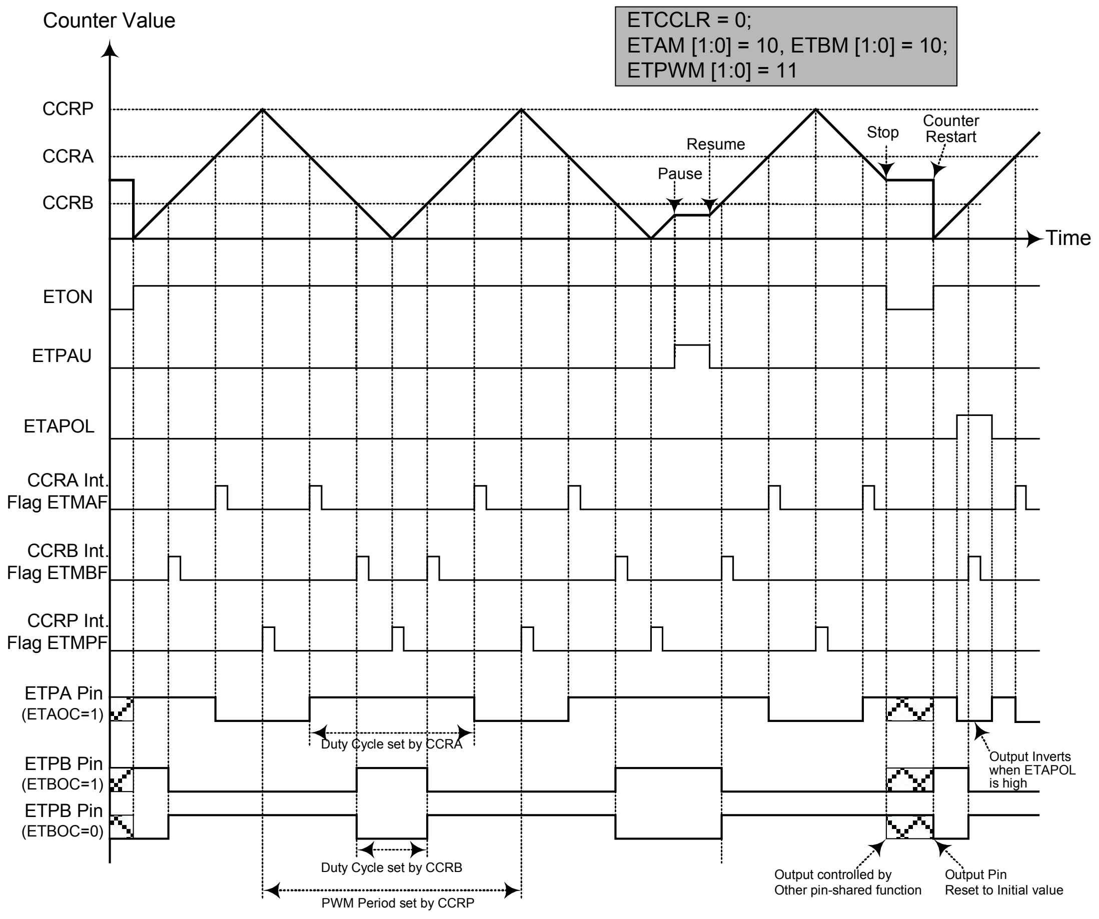
2. The internal PWM function continues running even when ETAIO [1:0] (or ETBIO [1:0]) = 00 or 01

3. CCRA controls the ETPA PWM duty and CCRB controls the ETPB PWM duty



ETM PWM Mode – Edge Aligned

Note: 1. Here ETCCLR=1 therefore CCRA clears the counter and determines the PWM period
2. The internal PWM function continues running even when ETBIO [1:0] = 00 or 01
3. The CCRA controls the ETPB PWM period and CCRB controls the ETPB PWM duty
4. Here the ETPA pin should not be used as a TM output pin.



ETM PWM Mode – Centre Aligned

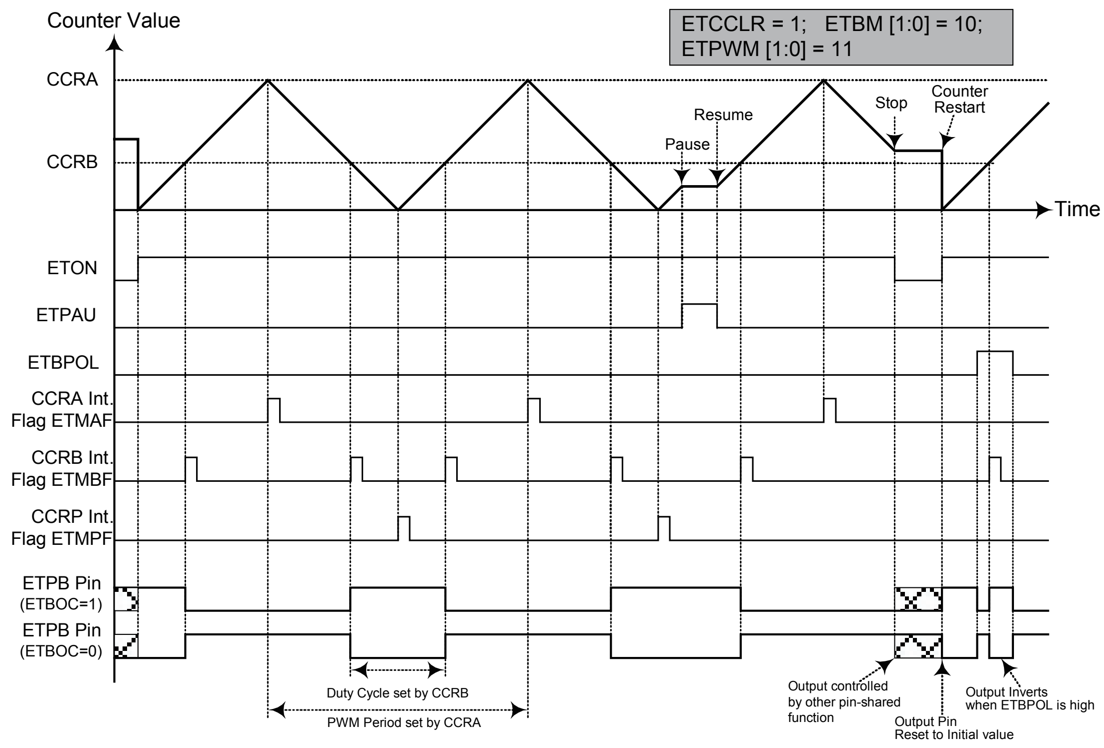
Note: 1. Here ETCCLR=0 therefore CCRP clears the counter and determines the PWM period

2. ETPWM [1:0] =11 therefore the PWM is centre aligned

3. The internal PWM function continues running even when ETAIO [1:0] (or ETBIO [1:0]) = 00 or 01

4. CCRA controls the ETPA PWM duty and CCRB controls the ETPB PWM duty

5. CCRP will generate an interrupt request when the counter decrements to its zero value



ETM PWM Mode – Centre Aligned

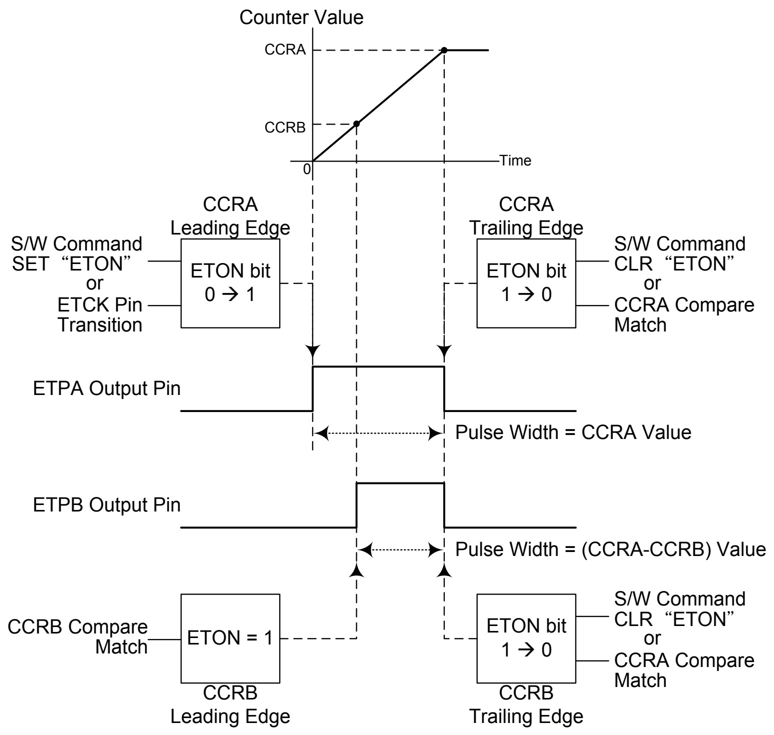
Note: 1. Here ETCCLR =1 therefore CCRA clears the counter and determines the PWM period
2. ETPWM [1:0] =11 therefore the PWM is centre aligned
3. The internal PWM function continues running even when ETBIO [1:0] = 00 or 01
4. CCRA controls the ETPB PWM period and CCRB controls the ETPB PWM duty
5. CCRP will generate an interrupt request when the counter decrements to its zero value

Single Pulse Output Mode

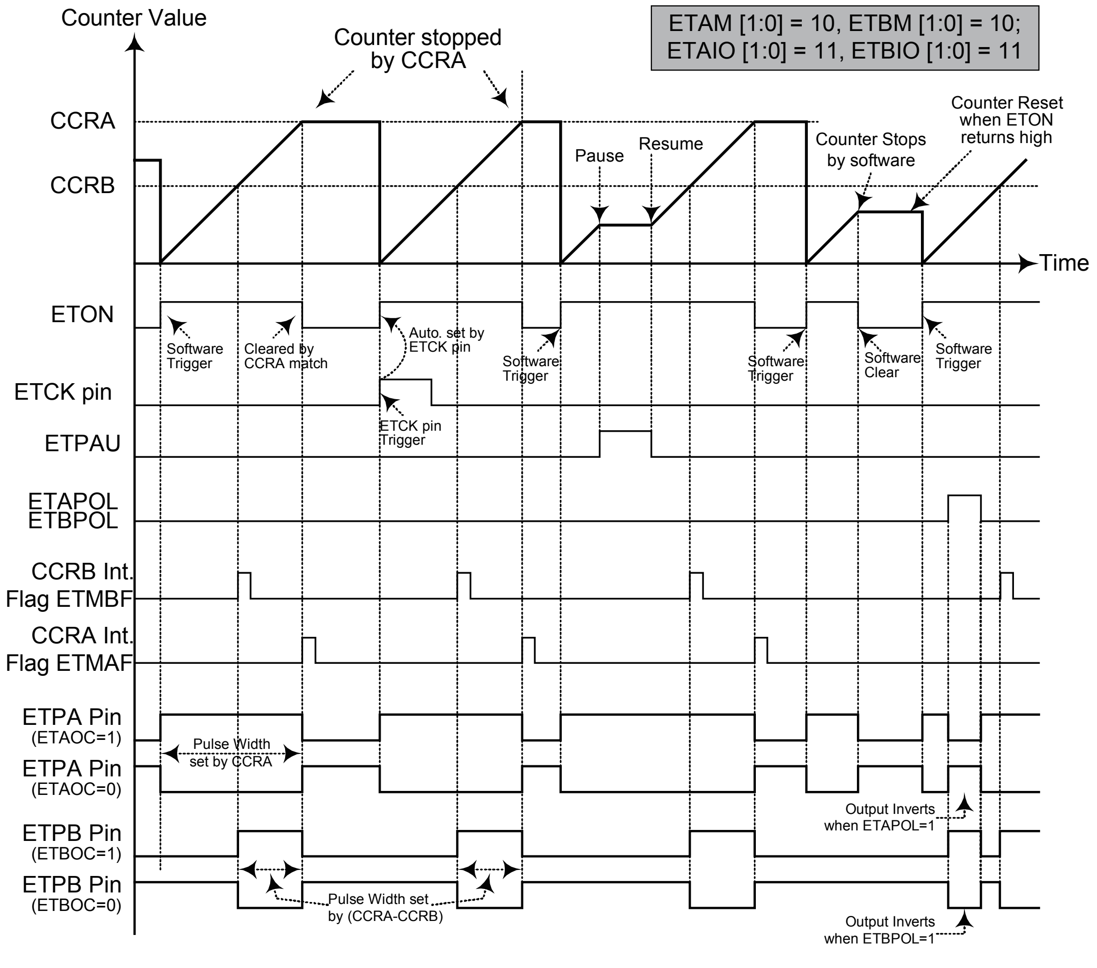
To select this mode, the required bit pairs, ETAM1, ETAM0 and ETBM1, ETBM0 should be set to 10 respectively and also the corresponding ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits should be set to 11 respectively. The Single Pulse Output Mode, as the name suggests, will generate a single shot pulse on the TM output pin.

The trigger for the pulse ETPA output leading edge is a low to high transition of the ETON bit, which can be implemented using the application program. The trigger for the pulse ETPB output leading edge is a compare match from Comparator B, which can be implemented using the application program. However in the Single Pulse Mode, the ETON bit can also be made to automatically change from low to high using the external ETCK pin, which will in turn initiate the Single Pulse output of ETPA. When the ETON bit transitions to a high level, the counter will start running and the pulse leading edge of ETPA will be generated. The ETON bit should remain high when the pulse is in its active state. The generated pulse trailing edge of ETPA and ETPB will be generated when the ETON bit is cleared to zero, which can be implemented using the application program or when a compare match occurs from Comparator A.

However a compare match from Comparator A will also automatically clear the ETON bit and thus generate the Single Pulse output trailing edge of ETPA and ETPB. In this way the CCRA value can be used to control the pulse width of ETPA. The (CCRA-CCRB) value can be used to control the pulse width of ETPB. A compare match from Comparator A and Comparator B will also generate TM interrupts. The counter can only be reset back to zero when the ETON bit changes from low to high when the counter restarts. In the Single Pulse Mode CCRP is not used. The ETCCLR bit is also not used.



Single Pulse Generation



ETM – Single Pulse Mode

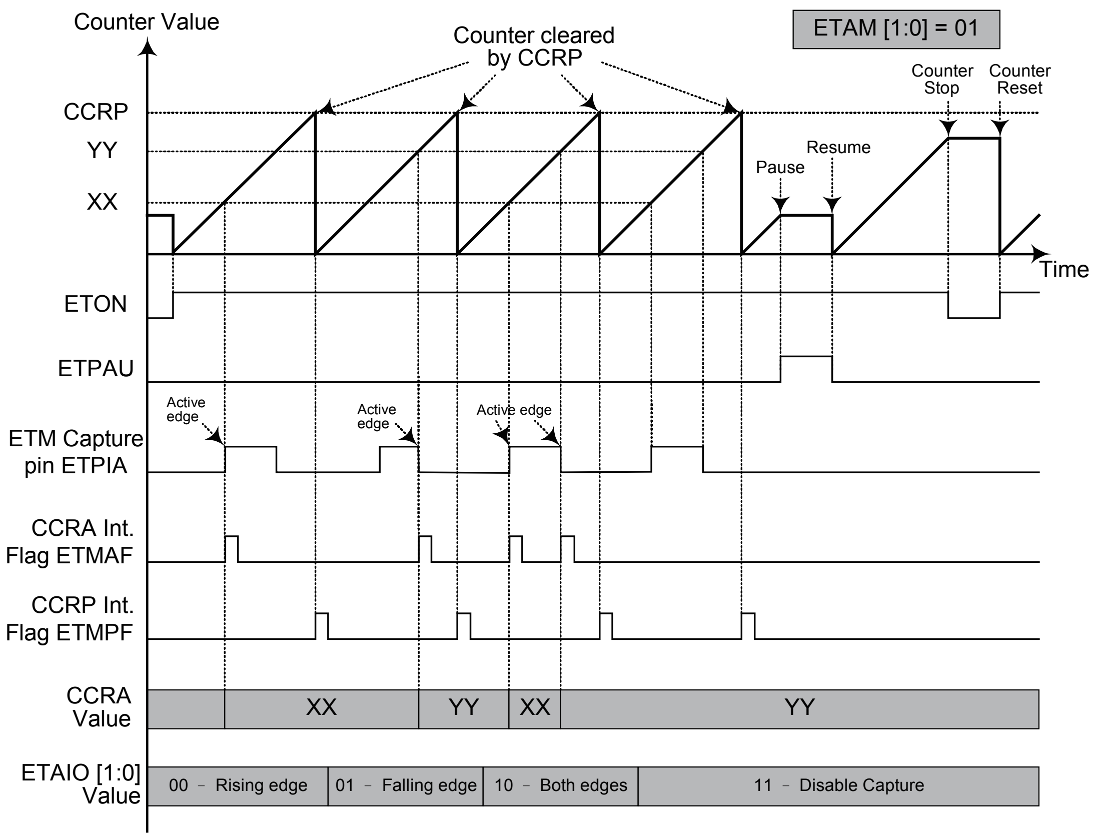
Note: 1. Counter stopped by CCRA
2. CCRP is not used
3. The pulse triggered by the ETCK pin or by setting the ETON bit high
4. A ETCK pin active edge will automatically set the ETON bit high.
5. In the Single Pulse Mode, ETAIO [1:0] and ETBIO [1:0] must be set to "11" and can not be changed

Capture Input Mode

To select this mode bits ETAM1, ETAM0 and ETBM1, ETBM0 in the ETMC1 and ETMC2 registers should be set to 01 respectively. This mode enables external signals to capture and store the present value of the internal counter and can therefore be used for applications such as pulse width measurements. The external signal is supplied on the ETPIA and ETPIB pins, whose active edge can be either a rising edge, a falling edge or both rising and falling edges; the active edge transition type is selected using the ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits in the ETMC1 and ETMC2 registers. The counter is started when the ETON bit changes from low to high which is initiated using the application program.

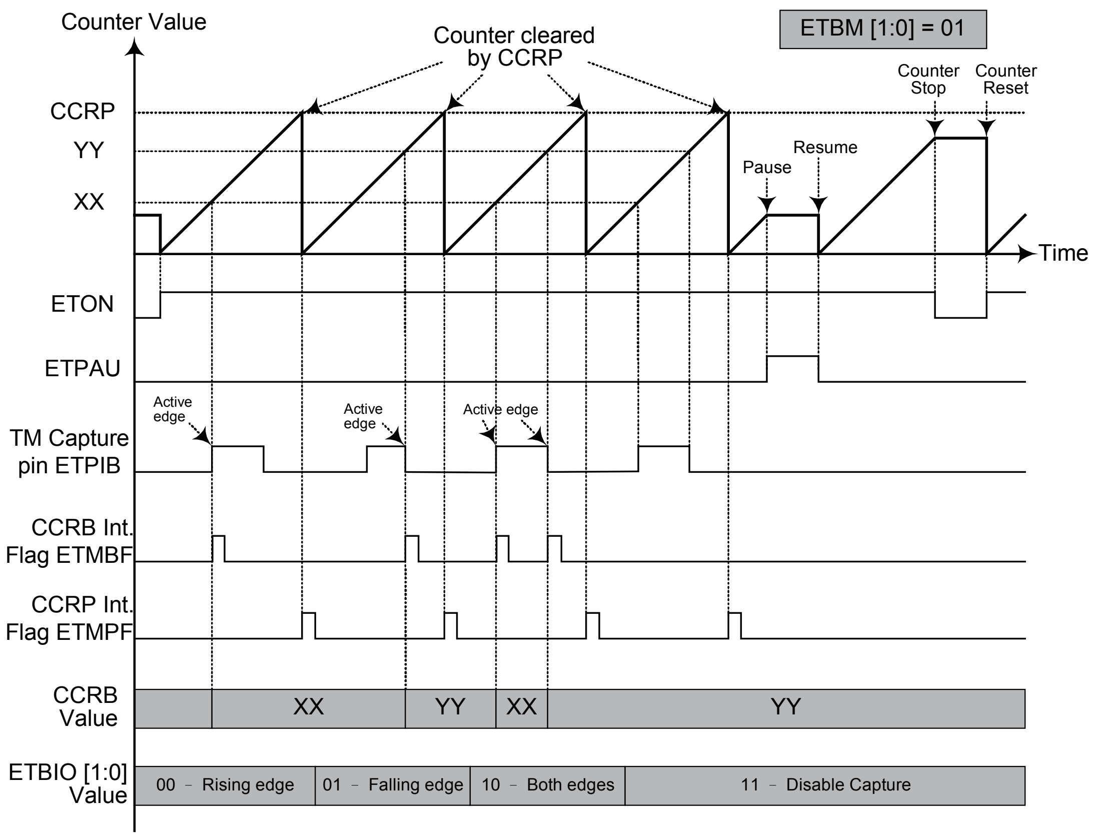
When the required edge transition appears on the ETPIA and ETPIB pins the present value in the counter will be latched into the CCRA and CCRB registers and a TM interrupt generated. Irrespective of what events occur on the ETPIA and ETPIB pins the counter will continue to free run until the ETON bit changes from high to low. When a CCRP compare match occurs the counter will reset back to zero; in this way the CCRP value can be used to control the maximum counter value. When a CCRP compare match occurs from Comparator P, a TM interrupt will also be generated. Counting the number of overflow interrupt signals from the CCRP can be a useful method in measuring long pulse widths. The ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits can select the active trigger edge on the ETPIA and ETPIB pins to be a rising edge, falling edge or both edge types. If the ETAIO1, ETAIO0 and ETBIO1, ETBIO0 bits are both set high, then no capture operation will take place irrespective of what happens on the ETPIA and ETPIB pins, however it must be noted that the counter will continue to run.

Note that the ETCCLR, ETAOC, ETBOC, ETAPOL and ETBPOL bits are not used in this mode.



ETM CCRA Capture Input Mode

Note: 1. ETAM [1:0] = 01 and active edge set by the ETAIO [1:0] bits
2. The TM Capture input pin active edge transfers the counter value to CCRA
3. ETCCLR bit not used
4. No output function – ETAOC and ETAPOL bits not used
5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero



ETM CCRB Capture Input Mode

Note: 1. ETBM [1:0] = 01 and active edge set by the ETBIO [1:0] bits
2. The TM Capture input pin active edge transfers the counter value to CCRB
3. ETCCLR bit not used
4. No output function -- ETBOC and ETBPOL bits not used
5. CCRP determines the counter value and the counter has a maximum count value when CCRP is equal to zero.

Software LCD Driver

The device has the capability of driving external LCD panels. The segment pins for LCD driving, SEG0~SEG23, are pin shared with certain pins on the PA~PD ports. The LCD signals, SEG0~SEG23, are generated using the application program.

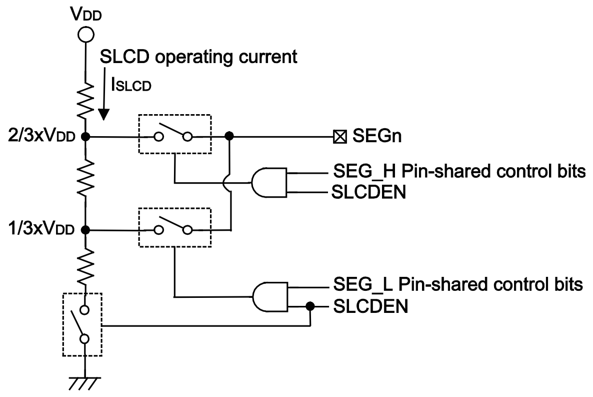
LCD Operation

An external LCD panel can be driven using this device by configuring the specific pins as segment or common pins. The LCD driver function is controlled using the SLCDC register which in addition to controlling the overall on/off function also controls the bias current selection function. This enables the LCD SEG driver to generate the necessary voltage levels for LCD 1/3 bias operation.

The SLCDEN bit in the SLCDC register is the overall master control for the software LCD driver. However, this bit is used to generate the LCD bias voltage levels and enable the specific SEG pins used for LCD driving together with the corresponding pin shared control bits. Note that the I/O Port ontrol register does not need to first setup the pins as outputs to enable the LCD driver operation. When the SLCDEN bit is set to 1 and the SEGn pin is configured to output a SEG_H level by the relevant pin shared control bits, the specific SEGn pin will output a 2/3*V_{DD} voltage. However, the specific SEGn pin will output a 1/3*V_{DD} voltage if the corresponding pin shared control bits set this SEGn pin to output a SEG_L level and the SLCDEN bit is high. Refer to the "Pin-shared Functions" section for detailed pin-shared control definitions.

SLCDEN	Pin shared Control Bits	Pin Function	Pin Voltage Level
1	10	SEG_L	1/3*V_{DD}
1	11	SEG_H	2/3*V_{DD}

SEG Pin Output Control



SLCD Bias and Pin Control

LCD Bias Control

The software LCD driver enables a range of selections to be provided to suit the requirement of the LCD panel which is being used. The bias current choice is implemented using the ISEL1 and ISEL0 bits in the SLCDC register.

SLCDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	ISEL1	ISEL0	SLCDEN	—	—	—	—
R/W	—	R/W	R/W	R/W	—	—	—	—
POR	—	0	0	0	—	—	—	—

Bit 7 Unimplemented, read as "0"

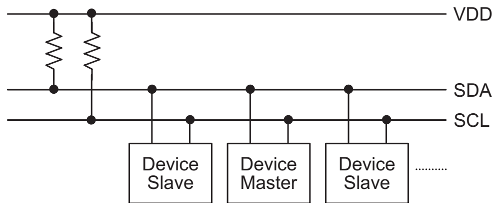
Bit 6~5 **ISEL1~ISEL0:** SLCD Bias Current Select
00: 8μA
01: 16μA
10: 50μA
11: 100μA

Bit 4 **SLCDEN:** SLCD function Enable control
0: disable
1: enable

Bit 3~0 Unimplemented, read as "0"

I²C Interface

The I²C interface is used to communicate with external peripheral devices such as sensors, EEPROM memory etc. Originally developed by Philips, it is a two line low speed serial interface for synchronous serial data transfer. The advantage of only two lines for communication, relatively simple communication protocol and the ability to accommodate multiple devices on the same bus has made it an extremely popular interface type for many applications.



I²C Master Slave Bus Connection

I²C Interface Operation

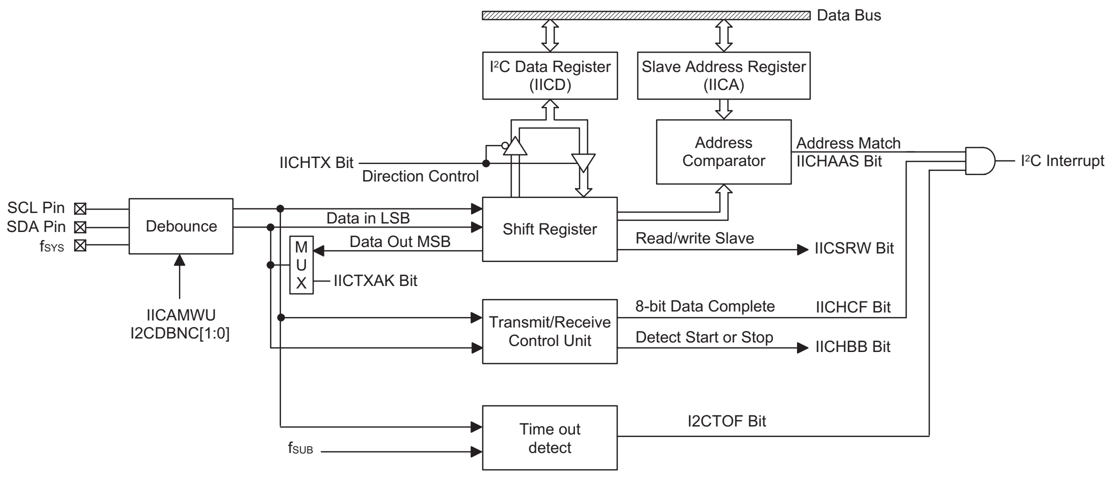
The I²C serial interface is a two line interface, a serial data line, SDA, and serial clock line, SCL. As many devices may be connected together on the same bus, their outputs are both open drain types. For this reason it is necessary that external pull-high resistors are connected to these outputs. This device provides an internal pull-high resistor which is connected on the SDA or SCL line respectively if the corresponding pull-high function control is enabled. Note that no chip select line exists, as each device on the I²C bus is identified by a unique address which will be transmitted and received on the I²C bus.

When two devices communicate with each other on the bidirectional I²C bus, one is known as the master device and one as the slave device. Both master and slave can transmit and receive data, however, it is the master device that has overall control of the bus. For these devices, which only operates in slave mode, there are two methods of transferring data on the I²C bus, the slave transmit mode and the slave receive mode.

The I2CDBNC1 and I2CDBNC0 bits determine the debounce time of the I^{2}C interface. This uses the system clock to in effect add a debounce time to the external clock to reduce the possibility of glitches on the clock line causing erroneous operation. The debounce time, if selected, can be chosen to be either 2 or 4 system clocks. To achieve the required I^{2}C data transfer speed, there exists a relationship between the system clock, f_{SYS}, and the I^{2}C debounce time. For either the I^{2}C Standard or Fast mode operation, users must take care of the selected system clock frequency and the configured debounce time to match the criterion shown in the following table.

I^{2}C Debounce Time Selection	I^{2}C Standard Mode (100kHz)	I^{2}C Fast Mode (400kHz)
No debounce	f_{SYS} > 2MHz	f_{SYS} > 5MHz
2 system clock debounce	f_{SYS} > 4MHz	f_{SYS} > 10MHz
4 system clock debounce	f_{SYS} > 8MHz	f_{SYS} > 20MHz

I^{2}C Minimum f_{SYS} Frequency



I^{2}C Block Diagram

I^{2}C Registers

There are three control registers associated with the I^{2}C bus, IICC0, IICC1 and I2CTOC, one data register, IICD, and one address register, IICA. The IICD register, which is shown in the I^{2}C block diagram, is used to store the data being transmitted and received on the I^{2}C bus. Before the microcontroller writes data to the I^{2}C bus, the actual data to be transmitted must be placed in the IICD register. After the data is received from the I^{2}C bus, the microcontroller can read it from the IICD register. Any transmission or reception of data from the I^{2}C bus must be made via the IICD register. The IICA register is used to store the I^{2}C slave address.

Name	Bit							
	7	6	5	4	3	2	1	0
IICC0	—	—	—	—	I2CDBNC1	I2CDBNC0	IICEN	—
IICC1	IICHCF	IICHAAS	IICHBB	IICHTX	IICTXAK	IICSRW	IICAMWU	IICRXAK
IICD	D7	D6	D5	D4	D3	D2	D1	D0
IICA	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0
I2CTOC	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0

I^{2}C Register List

IICC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	I2CDBNC1	I2CDBNC0	IICEN	—
R/W	—	—	—	—	R/W	R/W	R/W	—
POR	—	—	—	—	0	0	0	—

Bit 7~4 unimplemented, read as "0"

Bit 3~2 **I2CDBNC1~I2CDBNC0:** I^{2}C Debounce Time Selection
00: No debounce
01: 2 system clock debounce
1x: 4 system clock debounce

Bit 1 **IICEN:** I^{2}C interface enable control
0: Disable
1: Enable

Bit 0 unimplemented, read as "0"

IICC1 Register

Bit	7	6	5	4	3	2	1	0
Name	IICHCF	IICHAAS	IICHBB	IICHTX	IICTXAK	IICSRW	IICAMWU	IICRXAK
R/W	R	R	R	R/W	R/W	R	R/W	R
POR	1	0	0	0	0	0	0	1

Bit 7 **IICHCF:** I^{2}C Bus data transfer completion flag
0: Data is being transferred
1: Completion of an 8-bit data transfer
The IICHCF flag is the data transfer flag. This flag will be zero when data is being transferred. Upon completion of an 8-bit data transfer the flag will go high and an interrupt will be generated.

Bit 6 **IICHAAS:** I^{2}C Bus address match flag
0: Not address match
1: Address match
The IICHAAS flag is the address match flag. This flag is used to determine if the slave device address is the same as the master transmit address. If the addresses match then this bit will be high, if there is no match then the flag will be low.

Bit 5 **IICHBB:** I^{2}C Bus busy flag
0: I^{2}C Bus is not busy
1: I^{2}C Bus is busy
The IICHBB flag is the I^{2}C busy flag. This flag will be "1" when the I^{2}C bus is busy which will occur when a START signal is detected. The flag will be set to "0" when the bus is free which will occur when a STOP signal is detected.

Bit 4 **IICHTX:** Select I^{2}C slave device is transmitter or receiver
0: Slave device is the receiver
1: Slave device is the transmitter

Bit 3 **IICTXAK:** I^{2}C Bus transmit acknowledge flag
0: Slave send acknowledge flag
1: Slave do not send acknowledge flag
The IICTXAK bit is the transmit acknowledge flag. After the slave device receipt of 8-bits of data, this bit will be transmitted to the bus on the 9th clock from the slave device. The slave device must always set IICTXAK bit to "0" before further data is received.

Bit 2 **IICSRW:** I^{2}C Slave Read/Write flag
0: Slave device should be in receive mode
1: Slave device should be in transmit mode

The IICSRW flag is the I^{2}C Slave Read/Write flag. This flag determines whether the master device wishes to transmit or receive data from the I^{2}C bus. When the transmitted address and slave address is match, that is when the IICHAAS flag is set high, the slave device will check the IICSRW flag to determine whether it should be in transmit mode or receive mode. If the IICSRW flag is high, the master is requesting to read data from the bus, so the slave device should be in transmit mode. When the IICSRW flag is zero, the master will write data to the bus, therefore the slave device should be in receive mode to read this data.

Bit 1 **IICAMWU:** I^{2}C Address Match Wake-up Control
0: Disable
1: Enable - must be cleared by the application program after wake-up

This bit should be set to 1 to enable the I^{2}C address match wake up from the SLEEP or IDLE Mode. If the IICAMWU bit has been set before entering either the SLEEP or IDLE mode to enable the I^{2}C address match wake up, then this bit must be cleared by the application program after wake-up to ensure correction device operation.

Bit 0 **IICRXAK:** I^{2}C Bus Receive acknowledge flag
0: Slave receive acknowledge flag
1: Slave does not receive acknowledge flag

The IICRXAK flag is the receiver acknowledge flag. When the IICRXAK flag is "0", it means that a acknowledge signal has been received at the 9th clock, after 8 bits of data have been transmitted. When the slave device in the transmit mode, the slave device checks the IICRXAK flag to determine if the master receiver wishes to receive the next byte. The slave transmitter will therefore continue sending out data until the IICRXAK flag is "1". When this occurs, the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I^{2}C Bus.

IICD Register

Bit	7	6	5	4	3	2	1	0
Name	D7	D6	D5	D4	D3	D2	D1	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 I^{2}C Bus data bit 7~bit 0

The IICD register is used to store the data being transmitted and received. Before the device writes data to the I^{2}C bus, the actual data to be transmitted must be placed in the IICD register. After the data is received from the I^{2}C bus, the device can read it from the IICD register. Any transmission or reception of data from the I^{2}C bus must be made via the IICD register.

IICA Register

Bit	7	6	5	4	3	2	1	0
Name	IICA6	IICA5	IICA4	IICA3	IICA2	IICA1	IICA0	D0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7~1 IICA6~IICA0: I^{2}C Slave address bit 6~bit 0

The IICA register is the location where the 7-bit slave address of the I^{2}C slave device is stored. When a master device, which is connected to the I^{2}C bus, sends out an address, which matches the slave address in the IICA register, the slave device will be selected.

Bit 0 Undefined bit

This bit can be read or written by user software program.

I2CTOC Register

Bit	7	6	5	4	3	2	1	0
Name	I2CTOEN	I2CTOF	I2CTOS5	I2CTOS4	I2CTOS3	I2CTOS2	I2CTOS1	I2CTOS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **I2CTOEN:** I^{2}C interface Time-out control
0: Disable
1: Enable

Bit 6 **I2CTOF:** I^{2}C interface Time-out flag
0: No time-out occurred
1: Time-out occurred
The I2CTOF flag is set by the time-out circuitry when the time-out event occurs and cleared by software program.

Bit 5~0 **I2CDBNC1~I2CDBNC0:** I^{2}C interface Time-out period selection
I^{2}C interface Time-out period = $\frac{(I2CTOS[5:0]+1)x32}{f_{SUB}}$
The I^{2}C Time-out circuitry is driven by the (f_{SUB}/32) clock and the tinme-out period is calculated using the above formula.

I^{2}C Bus Communication

Communication on the I^{2}C bus requires four separate steps, a START signal, a slave device address transmission, a data transmission and finally a STOP signal. When a START signal is placed on the I^{2}C bus, all devices on the bus will receive this signal and be notified of the imminent arrival of data on the bus. The first seven bits of the data will be the slave address with the first bit being the MSB. If the address of the slave device matches that of the transmitted address, the IICHAAS bit in the IICC1 register will be set and an I^{2}C interrupt will be generated. After entering the interrupt service routine, the slave device must first check the condition of the IICHAAS bit to determine whether the interrupt source originates from an address match or from the completion of an 8-bit data transfer. During a data transfer, note that after the 7-bit slave address has been transmitted, the following bit, which is the 8th bit, is the read/write bit whose value will be placed in the IICSRW bit. This bit will be checked by the slave device to determine whether to go into transmit or receive mode. Before any transfer of data to or from the I^{2}C bus, the microcontroller must initialise the bus. The following are steps to achieve this:

- Step 1
  Configure the corresponding I/O pins as the I^{2}C interface pins, SDA and SCL, using the pin-shared control bits and set the IICEN bit high to enable the I^{2}C interface function.
- Step 2
  Write the slave address of the device to the I^{2}C bus address register IICA.
- Step 3
  Set the corresponding I^{2}C interrupt enable bit in the interrupt control register to enable the I^{2}C interrupt.

I^{2}C Bus Start Signal

The START signal can only be generated by the master device connected to the I^{2}C bus and not by the slave device. This START signal will be detected by all devices connected to the I^{2}C bus. When detected, this indicates that the I^{2}C bus is busy and therefore the IICHBB bit will be set. A START condition occurs when a high to low transition on the SDA line takes place when the SCL line remains high.

I^{2}C Slave Address

The transmission of a START signal by the master will be detected by all devices on the I^{2}C bus. To determine which slave device the master wishes to communicate with, the address of the slave device will be sent out immediately following the START signal. All slave devices, after receiving this 7-bit address data, will compare it with their own 7-bit slave address. If the address sent out by the master matches the internal address of the microcontroller slave device, then an internal I^{2}C bus interrupt signal will be generated. The next bit following the address, which is the 8th bit, defines the read/write status and will be saved to the IICSRW bit of the IICC1 register. The slave device will then transmit an acknowledge bit, which is a low level, as the 9th bit. The slave device will also set the status flag IICHAAS when the addresses match.

As an I^{2}C bus interrupt can come from three sources, when the program enters the interrupt subroutine, the I2CTOF bit should first be checked to see whether the I^{2}C interface time-out event occurs. If there is no I^{2}C time-out event occurs, then the IICHAAS bit should be examined to see whether the interrupt source has come from a matching slave address or from the completion of a data byte transfer. When a slave address is matched, the devices must be placed in either the transmit mode and then write data to the IICD register, or in the receive mode where it must implement a dummy read from the IICD register to release the SCL line.

I^{2}C Read/Write Signal

The IICSRW bit in the IICC1 register defines whether the slave device wishes to read data from the I^{2}C bus or write data to the I^{2}C bus. The slave device should examine this bit to determine if it is to be a transmitter or a receiver. If the IICSRW flag is "1" then this indicates that the master device wishes to read data from the I^{2}C bus, therefore the slave device must be setup to send data to the I^{2}C bus as a transmitter. If the IICSRW flag is "0" then this indicates that the master wishes to send data to the I^{2}C bus, therefore the slave device must be setup to read data from the I^{2}C bus as a receiver.

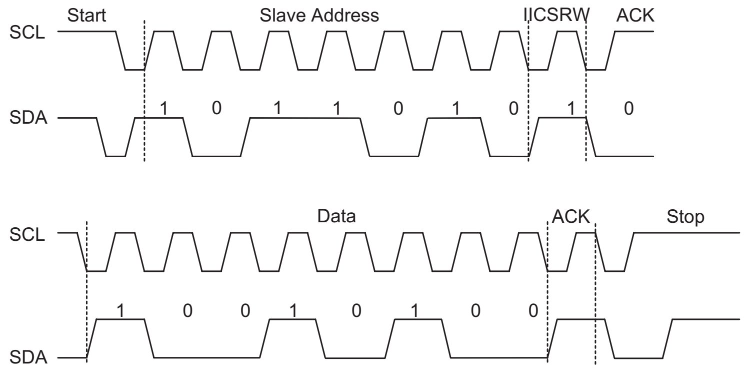
I^{2}C Slave Address Acknowledge Signal

After the master has transmitted a calling address, any slave device on the I^{2}C bus, whose own internal address matches the calling address, must generate an acknowledge signal. The acknowledge signal will inform the master that a slave device has accepted its calling address. If no acknowledge signal is received by the master then a STOP signal must be transmitted by the master to end the communication. When the IICHAAS flag is high, the addresses have matched and the slave device must check the IICSRW flag to determine if it is to be a transmitter or a receiver. If the IICSRW flag is high, the slave device should be setup to be a transmitter so the IICHTX bit in the IICC1 register should be set to "1". If the IICSRW flag is low, then the microcontroller slave device should be setup as a receiver and the IICHTX bit in the IICC1 register should be set to "0".

I²C Data/Acknowledge Signal

The transmitted data is 8-bits wide and is transmitted after the slave device has acknowledged receipt of its slave address. The order of serial bit transmission is the MSB first and the LSB last. After receipt of 8-bits of data, the receiver must transmit an acknowledge signal, level "0", before it can receive the next data byte. If the slave transmitter does not receive an acknowledge bit signal from the master receiver, then the slave transmitter will release the SDA line to allow the master to send a STOP signal to release the I²C Bus. The corresponding data will be stored in the IICD register. If setup as a transmitter, the slave device must first write the data to be transmitted into the IICD register. If setup as a receiver, the slave device must read the transmitted data from the IICD register.

When the slave receiver receives the data byte, it must generate an acknowledge bit, known as IICTXAK, on the 9th clock. The slave device, which is setup as a transmitter will check the IICRXAK bit in the IICC1 register to determine if it is to send another data byte, if not then it will release the SDA line and await the receipt of a STOP signal from the master.

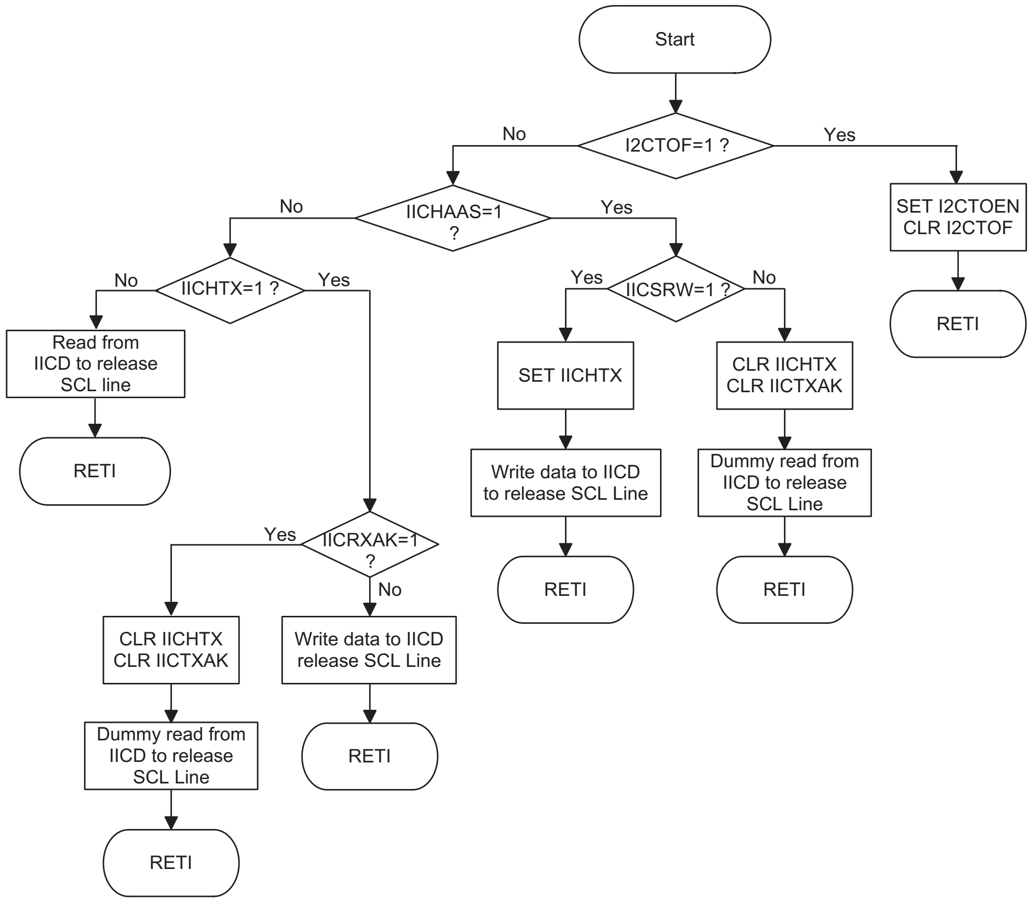


S=Start (1 bit)
SA=Slave Address (7 bits)
SR=IICSRW bit (1 bit)
M=Slave device send acknowledge bit (1 bit)
D=Data (8 bits)
A=ACK (IICRXAK bit for transmitter, IICTXAK bit for receiver 1 bit)
P=Stop (1 bit)

S	SA	SR	M	D	A	D	A	--------	S	SA	SR	M	D	A	D	A	--------	P

Note: When a slave address is matched, the devices must be placed in either the transmit mode and then write data to the IICD register, or in the receive mode where it must implememt a dummy read from the IICD register to release the SCL line.

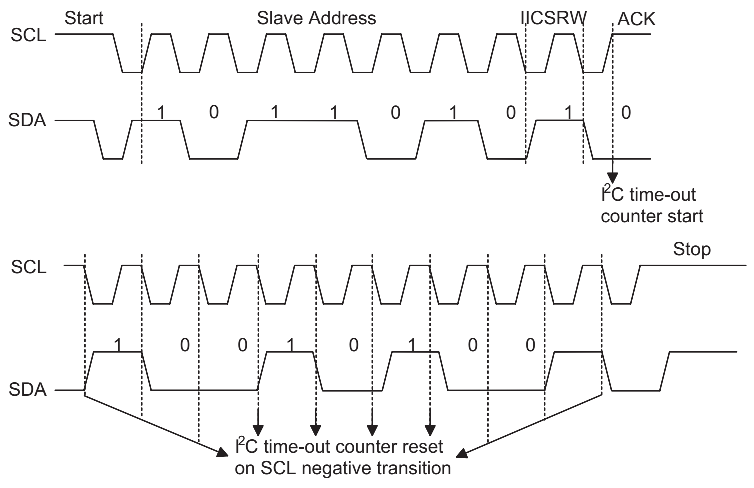
I²C Communication Timing Diagram


Start
I2CTOF=1 ?
No
Yes
SET I2CTOEN
CLR I2CTOF
RETI
IICHAAS=1 ?
No
Yes
IICHTX=1 ?
No
Yes
Read from IICD to release SCL line
RETI
IICSRW=1 ?
Yes
No
SET IICHTX
Write data to IICD to release SCL Line
RETI
CLR IICHTX
CLR IICTXAK
Dummy read from IICD to release SCL Line
RETI
IICRXAK=1 ?
Yes
No
CLR IICHTX
CLR IICTXAK
Dummy read from IICD to release SCL Line
RETI
Write data to IICD release SCL Line
RETI

I²C Bus ISR Flow Chart

I²C Timing-out Control

In order to reduce the I²C lockup problem due to reception of erroneous clock sources, a time-out function is provided. If the clock source connected to the I²C bus is not received for a while, then the I²C circuitry and registers will be reset after a certain time-out period. The time-out counter starts to count on an I²C bus "START" & "address match"condition, and is cleared by an SCL falling edge. Before the next SCL falling edge arrives, if the time elapsed is greater than the time-out period specified by the I2CTOC register, then a time-out condition will occur. The time-out function will stop when an I²C "STOP" condition occurs.


Start
Slave Address
IICSRW
ACK
SCL
SDA
1 0 1 1 0 1 0 1 0
I²C time-out counter start
Stop
SCL
SDA
1 0 0 1 0 1 0 0
I²C time-out counter reset on SCL negative transition

I²C Time-out Diagram

When an I²C time-out counter overflow occurs, the counter will stop and the I2CTOEN bit will be cleared to zero and the I2CTOF bit will be set high to indicate that a time-out condition has occurred. The time-out condition will also generate an interrupt which uses the I²C interrrupt vector. When an I²C time-out occurs, the I²C internal circuitry will be reset and the registers will be reset into the following condition:

Register	After I²C Tiome-out
IICD, IICA, IICC0	No change
IICC1	Reset to POR condition

I²C Registers after Time-out

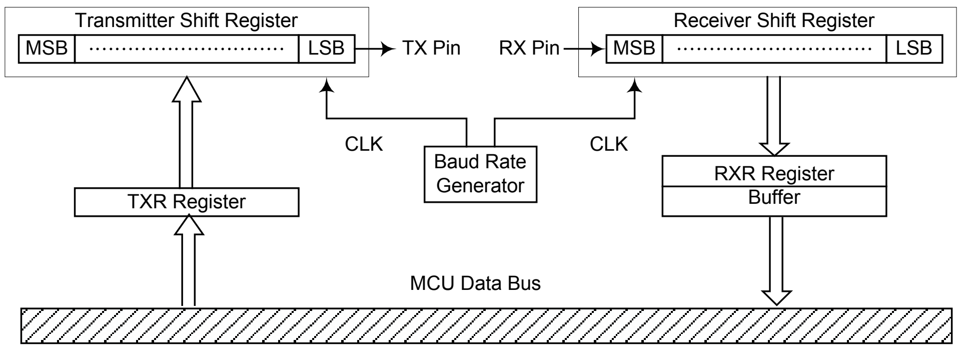
The I2CTOF flag can be cleared by the application program. There are 64 time-out period selections which can be selected using the I2CTOS bits in the I2CTOC register. The time-out duration is calculated by the formula: $((1\sim64) \times (32/f_{SUB}))$. This gives a time-out period which ranges from about 1ms to 64ms.

UART Interface

The device contains an integrated full-duplex asynchronous serial communications UART interface that enables communication with external devices that contain a serial interface. The UART function has many features and can transmit and receive data serially by transferring a frame of data with eight or nine data bits per transmission as well as being able to detect errors when the data is overwritten or incorrectly framed. The UART function possesses its own internal interrupt which can be used to indicate when a reception occurs or when a transmission terminates.

The integrated UART function contains the following features:

- Full-duplex, asynchronous communication
- 8 or 9 bits character length
- Even, odd or no parity options
- One or two stop bits
- Baud rate generator with 8-bit prescaler
- Parity, framing, noise and overrun error detection
- Support for interrupt on address detect (last character bit=1)
- Separately enabled transmitter and receiver
- 2-byte Deep FIFO Receive Data Buffer
- Transmit and receive interrupts
- Interrupts can be initialized by the following conditions:
    - Transmitter Empty
    - Transmitter Idle
    - Receiver Full
    - Receiver Overrun
    - Address Mode Detect



UART Data Transfer Scheme

UART External Pins

To communicate with an external serial interface, the internal UART has two external pins known as TX and RX. The TX pin is the UART transmitter pin, which can be used as a general purpose I/O or other pin-shared functional pin if the pin is not configured as a UART transmitter pin, which depends upon the corresponding pin-shared control configuration. Similarly, the RX pin is the UART receiver pin, which can also be used as a general purpose I/O pin or other pin-shared functional pin, if the pin is not configured as a receiver pin, which also depends upon the corresponding pin-shared control configuration. According to the pin-shared control configuration, if the TX or RX pin is selected as an UART interface pin, the internal pull-high resistor on the TX or RX pin will be disabled automatically.

UART Data Transfer Scheme

The block diagram shows the overall data transfer structure arrangement for the UART interface. The actual data to be transmitted from the MCU is first transferred to the TXR register by the application program. The data will then be transferred to the Transmit Shift Register from where it will be shifted out, LSB first, onto the TX pin at a rate controlled by the Baud Rate Generator. Only the TXR register is mapped onto the MCU Data Memory, the Transmit Shift Register is not mapped and is therefore inaccessible to the application program.

Data to be received by the UART is accepted on the external RX pin, from where it is shifted in, LSB first, to the Receiver Shift Register at a rate controlled by the Baud Rate Generator. When the shift register is full, the data will then be transferred from the shift register to the internal RXR register, where it is buffered and can be manipulated by the application program. Only the RXR register is mapped onto the MCU Data Memory, the Receiver Shift Register is not mapped and is therefore inaccessible to the application program.

It should be noted that the actual register for data transmission and reception, although referred to in the text, and in application programs, as separate TXR and RXR registers, only exists as a single shared register in the Data Memory. This shared register known as the TXR/RXR register is used for both data transmission and data reception.

UART Status and Control Registers

There are five control registers associated with the UART function. The USR, UCR1 and UCR2 registers control the overall function of the UART, while the BRG register controls the Baud rate. The actual data to be transmitted and received on the serial interface is managed through the TXR_RXR data register.

Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
USR	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
UCR1	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
UCR2	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
TXR_RXR	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
BRG	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0

UART Register List

TXR_RXR Register

The TXR_RXR register is the data register which is used to store the data to be transmitted on the TX pin or being received from the RX pin.

Bit	7	6	5	4	3	2	1	0
Name	TXRX7	TXRX6	TXRX5	TXRX4	TXRX3	TXRX2	TXRX1	TXRX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **TXRX7~TXRX0:** UART Transmit/Receive Data bits

USR Register

The USR register is the status register for the UART, which can be read by the program to determine the present status of the UART. All flags within the USR register are read only and further explanations are given below:

Bit	7	6	5	4	3	2	1	0
Name	PERR	NF	FERR	OERR	RIDLE	RXIF	TIDLE	TXIF
R/W	R	R	R	R	R	R	R	R
POR	0	0	0	0	1	0	1	1

Bit 7 **PERR:** Parity error flag
0: No parity error is detected
1: Parity error is detected
The PERR flag is the parity error flag. When this read only flag is "0", it indicates a parity error has not been detected. When the flag is "1", it indicates that the parity of the received word is incorrect. This error flag is applicable only if Parity mode (odd or even) is selected. The flag can also be cleared by a software sequence which involves a read to the status register USR followed by an access to the RXR data register.

Bit 6 **NF:** Noise flag
0: No noise is detected
1: Noise is detected
The NF flag is the noise flag. When this read only flag is "0", it indicates no noise condition. When the flag is "1", it indicates that the UART has detected noise on the receiver input. The NF flag is set during the same cycle as the RXIF flag but will not be set in the case of as overrun. The NF flag can be cleared by a software sequence which will involve a read to the status register USR followed by an access to the RXR data register.

Bit 5 **FERR:** Framing error flag
0: No framing error is detected
1: Framing error is detected
The FERR flag is the framing error flag. When this read only flag is "0", it indicates that there is no framing error. When the flag is "1", it indicates that a framing error has been detected for the current character. The flag can also be cleared by a software sequence which will involve a read to the status register USR followed by an access to the RXR data register.

Bit 4 **OERR:** Overrun error flag
0: No overrun error is detected
1: Overrun error is detected
The OERR flag is the overrun error flag which indicates when the receiver buffer has overflowed. When this read only flag is "0", it indicates that there is no overrun error. When the flag is "1", it indicates that an overrun error occurs which will inhibit further transfers to the RXR receive data register. The flag is cleared by a software sequence, which is a read to the status register USR followed by an access to the RXR data register.

Bit 3 **RIDLE:** Receiver status
0: data reception is in progress (data being received)
1: no data reception is in progress (receiver is idle)

The RIDLE flag is the receiver status flag. When this read only flag is "0", it indicates that the receiver is between the initial detection of the start bit and the completion of the stop bit. When the flag is "1", it indicates that the receiver is idle. Between the completion of the stop bit and the detection of the next start bit, the RIDLE bit is "1" indicating that the UART receiver is idle and the RX pin stays in logic high condition.

Bit 2 **RXIF:** Receive RXR data register status
0: RXR data register is empty
1: RXR data register has available data

The RXIF flag is the receive data register status flag. When this read only flag is "0", it indicates that the RXR read data register is empty. When the flag is "1", it indicates that the RXR read data register contains new data. When the contents of the shift register are transferred to the RXR register, an interrupt is generated if RIE=1 in the UCR2 register. If one or more errors are detected in the received word, the appropriate receive-related flags NF, FERR, and/or PERR are set within the same clock cycle. The RXIF flag is cleared when the USR register is read with RXIF set, followed by a read from the RXR register, and if the RXR register has no data available.

Bit 1 **TIDLE:** Transmission status
0: data transmission is in progress (data being transmitted)
1: no data transmission is in progress (transmitter is idle)

The TIDLE flag is known as the transmission complete flag. When this read only flag is "0", it indicates that a transmission is in progress. This flag will be set to "1" when the TXIF flag is "1" and when there is no transmit data or break character being transmitted. When TIDLE is equal to 1, the TX pin becomes idle with the pin state in logic high condition. The TIDLE flag is cleared by reading the USR register with TIDLE set and then writing to the TXR register. The flag is not generated when a data character or a break is queued and ready to be sent.

Bit 0 **TXIF:** Transmit TXR data register status
0: character is not transferred to the transmit shift register
1: character has transferred to the transmit shift register (TXR data register is empty)

The TXIF flag is the transmit data register empty flag. When this read only flag is "0", it indicates that the character is not transferred to the transmitter shift register. When the flag is "1", it indicates that the transmitter shift register has received a character from the TXR data register. The TXIF flag is cleared by reading the UART status register (USR) with TXIF set and then writing to the TXR data register. Note that when the TXEN bit is set, the TXIF flag bit will also be set since the transmit data register is not yet full.

UCR1 Register

The UCR1 register together with the UCR2 register are the two UART control registers that are used to set the various options for the UART function such as overall on/off control, parity control, data transfer bit length, etc. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	UARTEN	BNO	PREN	PRT	STOPS	TXBRK	RX8	TX8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R	W
POR	0	0	0	0	0	0	x	0

Bit 7 **UARTEN:** UART function enable control
0: disable UART. TX and RX pins can be I/O or other pin-shared functions
1: enable UART. TX and RX pins should be configured as UART pins

The UARTEN bit is the UART enable bit. When this bit is equal to "0", the UART will be disabled and the RX pin as well as the TX pin can be set as I/O or other pin-shared functions. When the bit is equal to "1", the UART will be enabled and the TX and RX pins will function as defined by the TXEN and RXEN enable control bits. When the UART is disabled, it will empty the buffer so any character remaining in the buffer will be discarded. In addition, the value of the baud rate counter will be reset. If the UART is disabled, all error and status flags will be reset. Also the TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF bits will be cleared, while the TIDLE, TXIF and RIDLE bits will be set. Other control bits in UCR1, UCR2 and BRG registers will remain unaffected. If the UART is active and the UARTEN bit is cleared, all pending transmissions and receptions will be terminated and the module will be reset as defined above. When the UART is re-enabled, it will restart in the same configuration.

Bit 6 **BNO:** Number of data transfer bits selection
0: 8-bit data transfer
1: 9-bit data transfer
This bit is used to select the data length format, which can have a choice of either 8-bit or 9-bit format. When this bit is equal to "1", a 9-bit data length format will be selected. If the bit is equal to "0", then an 8-bit data length format will be selected. If 9-bit data length format is selected, then bits RX8 and TX8 will be used to store the 9th bit of the received and transmitted data respectively.

Bit 5 **PREN:** Parity function enable control
0: Parity function is disabled
1: Parity function is enabled
This bit is the parity function enable bit. When this bit is equal to 1, the parity function will be enabled. If the bit is equal to 0, then the parity function will be disabled.

Bit 4 **PRT:** Parity type selection bit
0: Even parity for parity generator
1: Odd parity for parity generator
This bit is the parity type selection bit. When this bit is equal to 1, odd parity type will be selected. If the bit is equal to 0, then even parity type will be selected.

Bit 3 **STOPS:** Number of stop bits selection
0: One stop bit format is used
1: Two stop bits format is used
This bit determines if one or two stop bits are to be used. When this bit is equal to "1", two stop bits format are used. If the bit is equal to "0", then only one stop bit format is used.

Bit 2 **TXBRK:** Transmit break character
0: No break character is transmitted
1: Break characters transmit
The TXBRK bit is the Transmit Break Character bit. When this bit is equal to "0", there are no break characters and the TX pin operates normally. When the bit is equal to "1", there are transmit break characters and the transmitter will send logic zeros. When this bit is equal to "1", after the buffered data has been transmitted, the transmitter output is held low for a minimum of a 13-bit length and until the TXBRK bit is reset.

Bit 1 **RX8:** Receive data bit 8 for 9-bit data transfer format (read only)
This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the received data known as RX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

Bit 0 **TX8:** Transmit data bit 8 for 9-bit data transfer format (write only)
This bit is only used if 9-bit data transfers are used, in which case this bit location will store the 9th bit of the transmitted data known as TX8. The BNO bit is used to determine whether data transfers are in 8-bit or 9-bit format.

UCR2 Register

The UCR2 register is the second of the UART control registers and serves several purposes. One of its main functions is to control the basic enable/disable operation if the UART Transmitter and Receiver as well as enabling the various UART interrupt sources. The register also serves to control the baud rate speed, receiver wake-up function enable and the address detect function enable. Further explanation on each of the bits is given below:

Bit	7	6	5	4	3	2	1	0
Name	TXEN	RXEN	BRGH	ADDEN	WAKE	RIE	TIIE	TEIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **TXEN:** UART Transmitter enable control
0: UART Transmitter is disabled
1: UART Transmitter is enabled

The bit named TXEN is the Transmitter Enable Bit. When this bit is equal to "0", the transmitter will be disabled with any pending data transmissions being aborted. In addition the buffers will be reset. In this situation the TX pin can be used as an I/O or other pin-shared functional pin. If the TXEN bit is equal to "1" and the UARTEN bit is also equal to 1, the transmitter will be enabled and the TX pin will be controlled by the UART. Clearing the TXEN bit during a transmission will cause the data transmission to be aborted and will reset the transmitter. If this situation occurs, the TX pin can be used as an I/O or other pin-shared functional pin.

Bit 6 **RXEN:** UART Receiver enable control
0: UART Receiver is disabled
1: UART Receiver is enabled

The bit named RXEN is the Receiver Enable Bit. When this bit is equal to "0", the receiver will be disabled with any pending data receptions being aborted. In addition the receiver buffers will be reset. In this situation the RX pin can be used as an I/O or other pin-shared functional pin. If the RXEN bit is equal to "1" and the UARTEN bit is also equal to 1, the receiver will be enabled and the RX pin will be controlled by the UART. Clearing the RXEN bit during a reception will cause the data reception to be aborted and will reset the receiver. If this situation occurs, the RX pin can be used as an I/O or other pin-shared functional pin.

Bit 5 **BRGH:** Baud Rate speed selection
0: Low speed baud rate
1: High speed baud rate

The bit named BRGH selects the high or low speed mode of the Baud Rate Generator. This bit, together with the value placed in the baud rate register, BRG, controls the baud rate of the UART. If the bit is equal to 0, the low speed mode is selected.

Bit 4 **ADDEN:** Address detect function enable control
0: Address detection function is disabled
1: Address detection function is enabled

The bit named ADDEN is the address detection function enable control bit. When this bit is equal to 1, the address detection function is enabled. When it occurs, if the 8th bit, which corresponds to RX7 if BNO=0, or the 9th bit, which corresponds to RX8 if BNO=1, has a value of "1", then the received word will be identified as an address, rather than data. If the corresponding interrupt is enabled, an interrupt request will be generated each time the received word has the address bit set, which is the 8th or 9th bit depending on the value of the BNO bit. If the address bit known as the 8th or 9th bit of the received word is "0" with the address detection function being enabled, an interrupt will not be generated and the received data will be discarded.

Bit 3	**WAKE:** RX pin falling edge wake-up function enable control 0: RX pin wake-up function is disabled 1: RX pin wake-up function is enabled The bit enables or disables the receiver wake-up function. If this bit is equal to 1 and the device is in IDLE or SLEEP mode, a falling edge on the RX pin will wake up the device. If this bit is equal to 0 and the device is in IDLE or SLEEP mode, any edge transitions on the RX pin will not wake up the device.
Bit 2	**RIE:** Receiver interrupt enable control 0: Receiver related interrupt is disabled 1: Receiver related interrupt is enabled The bit enables or disables the receiver interrupt. If this bit is equal to 1 and when the receiver overrun flag OERR or received data available flag RXIF is set, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the OERR or RXIF flags.
Bit 1	**TIIE:** Transmitter Idle interrupt enable control 0: Transmitter idle interrupt is disabled 1: Transmitter idle interrupt is enabled The bit enables or disables the transmitter idle interrupt. If this bit is equal to 1 and when the transmitter idle flag TIDLE is set, due to a transmitter idle condition, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the TIDLE flag.
Bit 0	**TEIE:** Transmitter Empty interrupt enable control 0: Transmitter empty interrupt is disabled 1: Transmitter empty interrupt is enabled The bit enables or disables the transmitter empty interrupt. If this bit is equal to 1 and when the transmitter empty flag TXIF is set, due to a transmitter empty condition, the UART interrupt request flag will be set. If this bit is equal to 0, the UART interrupt request flag will not be influenced by the condition of the TXIF flag.

Baud Generator

To setup the speed of the serial data communication, the UART function contains its own dedicated baud rate generator. The baud rate is controlled by its own internal free running 8-bit timer, the period of which is determined by two factors. The first of these is the value placed in the BRG register and the second is the value of the BRGH bit within the UCR2 control register. The BRGH bit decides, if the baud rate generator is to be used in a high speed mode or low speed mode, which in turn determines the formula that is used to calculate the baud rate. The value in the BRG register, N, which is used in the following baud rate calculation formula determines the division factor. Note that N is the decimal value placed in the BRG register and has a range of between 0 and 255.

UCR2 BRGH Bit	0	1
Baud Rate (BR)	$\frac{f_{SYS}}{[64(N+1)]}$	$\frac{f_{SYS}}{[16(N+1)]}$

By programming the BRGH bit which allows selection of the related formula and programming the required value in the BRG register, the required baud rate can be setup. Note that because the actual baud rate is determined using a discrete value, N, placed in the BRG register, there will be an error associated between the actual and requested value. The following example shows how the BRG register value N and the error value can be calculated.

- **BRG Register**

Bit	7	6	5	4	3	2	1	0
Name	BRG7	BRG6	BRG5	BRG4	BRG3	BRG2	BRG1	BRG0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	x	x	x	x	x	x	x	x

"x": unknown

Bit 7~0 **BRG7~BRG0:** Baud Rate values

By programming the BRGH bit in the UCR2 register which allows selection of the related formula described above and programming the required value in the BRG register, the required baud rate can be setup.

Calculating the Baud Rate and Error Values

For a clock frequency of 4MHz, and with BRGH set to 0 determine the BRG register value N, the actual baud rate and the error value for a desired baud rate of 4800.

From the above table the desired baud rate BR $= \frac{f_{SYS}}{[64(N+1)]}$

Re-arranging this equation gives N $= \frac{f_{SYS}}{(BR \times 64)} - 1$

Giving a value for N $= \frac{4000000}{(4800 \times 64)} - 1 = 12.0208$

To obtain the closest value, a decimal value of 12 should be placed into the BRG register. This gives an actual or calculated baud rate value of BR$= \frac{4000000}{[64(12+1)]} = 4808$

Therefore the error is equal to $\frac{4808 - 4800}{4800} = 0.16\%$

The following tables show the actual values of baud rate and error values for the two values of BRGH.

Baud Rate K/BPS	Baud Rates for BRGH=0								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRGn	Kbaud	Error(%)	BRGn	Kbaud	Error(%)	BRGn	Kbaud	Error(%)
0.3	207	0.300	0.16	185	0.300	0.00	—	—	—
1.2	51	1.202	0.16	46	1.190	-0.83	92	1.203	0.23
2.4	25	2.404	0.16	22	2.432	1.32	46	2.380	-0.83
4.8	12	4.808	0.16	11	4.661	-2.90	22	4.863	1.32
9.6	6	8.929	-6.99	5	9.321	-2.90	11	9.332	-2.90
19.2	2	20.833	8.51	2	18.643	-2.90	5	18.643	-2.90
38.4	—	—	—	—	—	—	2	32.286	-2.90
57.6	0	62.500	8.51	0	55.930	-2.90	1	55.930	-2.90
115.2	—	—	—	—	—	—	0	111.859	-2.90

Baud Rates and Error Values for BRGH=0

Baud Rate K/BPS	Baud Rates for BRGH=1								
	f_{SYS}=4MHz			f_{SYS}=3.579545MHz			f_{SYS}=7.159MHz		
	BRGn	Kbaud	Error(%)	BRGn	Kbaud	Error(%)	BRGn	Kbaud	Error(%)
0.3	—	—	—	—	—	—	—	—	—
1.2	207	1.202	0.16	185	1.203	0.23	—	—	—
2.4	103	2.404	0.16	92	2.406	0.23	185	2.406	0.23
4.8	51	4.808	0.16	46	4.76	-0.83	92	4.811	0.23
9.6	25	9.615	0.16	22	9.727	1.32	46	9.520	-0.83
19.2	12	19.231	0.16	11	18.643	-2.90	22	18.643	1.32
38.4	6	35.714	-6.99	5	37.286	-2.90	11	37.286	-2.90
57.6	0	62.500	8.51	3	55.930	-2.90	7	55.930	-2.90
115.2	1	125	8.51	1	111.86	-2.90	3	111.86	-2.90
250	0	250	0	—	—	—	—	—	—

Baud Rates and Error Values for BRGH=0

UART Setup and Control

For data transfer, the UART function utilizes a non-return-to-zero, more commonly known as NRZ, format. This is composed of one start bit, eight or nine data bits and one or two stop bits. Parity is supported by the UART hardware and can be setup to be even, odd or no parity. For the most common data format, 8 data bits along with no parity and one stop bit, denoted as 8, N, 1, is used as the default setting, which is the setting at power-on. The number of data bits and stop bits, along with the parity, are setup by programming the corresponding BNO, PRT, PREN and STOPS bits in the UCR1 register. The baud rate used to transmit and receive data is setup using the internal 8-bit baud rate generator, while the data is transmitted and received LSB first. Although the transmitter and receiver of the UART are functionally independent, they both use the same data format and baud rate. In all cases stop bits will be used for data transmission.

Enabling/Disable the UART Interface

The basic on/off function of the internal UART function is controlled using the UARTEN bit in the UCR1 register. If the UARTEN, TXEN and RXEN bits are set, then these two UART pins will act as normal TX output pin and RX input pin respectively. If no data is being transmitted on the TX pin, then it will default to a logic high value.

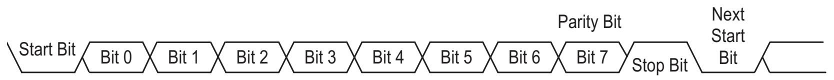
Clearing the UARTEN bit will disable the TX and RX pins and these two pins can be used as I/O or other pin-shared functional pins. When the UART function is disabled, the buffer will be reset to an empty condition, at the same time discarding any remaining residual data. Disabling the UART will also reset the enable control, the error and status flags with bits TXEN, RXEN, TXBRK, RXIF, OERR, FERR, PERR and NF being cleared while bits TIDLE, TXIF and RIDLE will be set. The remaining control bits in the UCR1, UCR2 and BRG registers will remain unaffected. If the UARTEN bit in the UCR1 register is cleared while the UART is active, then all pending transmissions and receptions will be immediately suspended and the UART will be reset to a condition as defined above. If the UART is then subsequently re-enabled, it will restart again in the same configuration.

Data, Parity and Stop Bit Selection

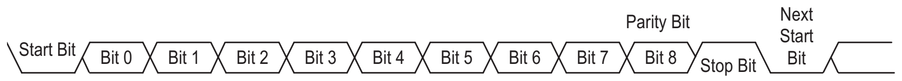
The format of the data to be transferred is composed of various factors such as data bit length, parity on/off, parity type, address bits and the number of stop bits. These factors are determined by the setup of various bits within the UCR1 register. The BNO bit controls the number of data bits which can be set to either 8 or 9. The PRT bit controls the choice if odd or even parity. The PREN bit controls the parity on/off function. The STOPS bit decides whether one or two stop bits are to be used. The following table shows various formats for data transmission. The address detect mode control bit identifies the frame as an address character. The number of stop bits, which can be either one or two, is independent of the data length.

Start Bit	Data Bits	Address Bits	Parity Bit	Stop Bit
Example of 8-bit Data Formats				
1	8	0	0	1
1	7	0	1	1
1	7	1	0	1
Example of 9-bit Data Formats				
1	9	0	0	1
1	8	0	1	1
1	8	1	0	1

The following diagram shows the transmit and receive waveforms for both 8-bit and 9-bit data formats.



8-Bit Data Format



9-Bit Data Format

UART Transmitter

Data word lengths of either 8 or 9 bits can be selected by programming the BNOn bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, needs to be stored in the TX8 bit in the UCR1 register. At the transmitter core lies the Transmitter Shift Register, more commonly known as the TSR, whose data is obtained from the transmit data register, which is known as the TXR register. The data to be transmitted is loaded into this TXR register by the application program. The TSR register is not written to with new data until the stop bit from the previous transmission has been sent out. As soon as this stop bit has been transmitted, the TSR can then be loaded with new data from the TXR register, if it is available. It should be noted that the TSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations. An actual transmission of data will normally be enabled when the TXEN bit is set, but the data will not be transmitted until the TXR register has been loaded with data and the baud rate generator has defined a shift clock source. However, the transmission can also be initiated by first loading data into the TXR register, after which the TXEN bit can be set. When a transmission of data begins, the TSR is normally empty, in which case a transfer to the TXR register will result in an immediate transfer to the TSR. If during a transmission the TXEN bit is cleared, the transmission will immediately cease and the transmitter will be reset. The TX output pin can then return to the I/O or other pin-shared function.

Transmitting Data

When the UARTn is transmitting data, the data is shifted on the TXn pin from the shift register, with the least significant bit LSB first. In the transmit mode, the TXRn register forms a buffer between the internal bus and the transmitter shift register. It should be noted that if 9-bit data format has been selected, then the MSB will be taken from the TX8n bit in the UCR1n register. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the TXEN bit to ensure that the UART transmitter is enabled and the TX pin is used as a UART transmitter pin.
- Access the USR register and write the data that is to be transmitted into the TXR register. Note that this step will clear the TXIF bit.

This sequence of events can now be repeated to send additional data. It should be noted that when TXIF=0, data will be inhibited from being written to the TXR register. Clearing the TXIF flag is always achieved using the following software sequence:

- A USR register access
- A TXR register write execution

The read-only TXIF flag is set by the UART hardware and if set indicates that the TXR register is empty and that other data can now be written into the TXR register without overwriting the previous data. If the TEIE bit is set, then the TXIF flag will generate an interrupt. During a data transmission, a write instruction to the TXR register will place the data into the TXR register, which will be copied to the shift register at the end of the present transmission. When there is no data transmission in progress, a write instruction to the TXR register will place the data directly into the shift register, resulting in the commencement of data transmission, and the TXIF bit being immediately set. When a frame transmission is complete, which happens after stop bits are sent or after the break frame, the TIDLE bit will be set. To clear the TIDLE bit the following software sequence is used:

- A USR register access
- A TXR register write execution

Note that both the TXIF and TIDLE bits are cleared by the same software sequence.

Transmitting Break

If the TXBRK bit is set, then the break characters will be sent on the next transmission. Break character transmission consists of a start bit, followed by 13xN "0" bits, where N=1, 2, etc. if a break character is to be transmitted, then the TXBRK bit must be first set by the application program and then cleared to generate the stop bits. Transmitting a break character will not generate a transmit interrupt. Note that a break condition length is at least 13 bits long. If the TXBRK bit is continually kept at a logic high level, then the transmitter circuitry will transmit continuous break characters. After the application program has cleared the TXBRK bit, the transmitter will finish transmitting the last break character and subsequently send out one or two stop bits. The automatic logic high at the end of the last break character will ensure that the start bit of the next frame is recognized.

UART Receiver

The UART is capable of receiving word lengths of either 8 or 9 bits can be selected by programming the BNO bit in the UCR1 register. When BNO bit is set, the word length will be set to 9 bits. In this case the 9th bit, which is the MSB, will be stored in the RX8 bit in the UCR1 register. At the receiver core lies the Receiver Shift Register more commonly known as the RSR. The data which is received on the RX external input pin is sent to the data recovery block. The data recovery block operating speed is 16 times that of the baud rate, while the main receive serial shifter operates at the baud rate. After the RX pin is sampled for the stop bit, the received data in RSR is transferred to the receive data register, if the register is empty. The data which is received on the external RX input pin is sampled three times by a majority detect circuit to determine the logic level that has been placed onto the RX pin. It should be noted that the RSR register, unlike many other registers, is not directly mapped into the Data Memory area and as such is not available to the application program for direct read/write operations.

Receiving Data

When the UART receiver is receiving data, the data is serially shifted in on the external RX input pin to the shift register, with the least significant bit LSB first. The RXR register is a two byte deep FIFO data buffer, where two bytes can be held in the FIFO while the third byte can continue to be received. Note that the application program must ensure that the data is read from RXR before the third byte has been completely shifted in, otherwise the third byte will be discarded and an overrun error OERR will be subsequently indicated. The steps to initiate a data transfer can be summarized as follows:

- Make the correct selection of the BNO, PRT, PREN and STOPS bits to define the required word length, parity type and number of stop bits.
- Setup the BRG register to select the desired baud rate.
- Set the RXEN bit to ensure that the UART receiver is enabled and the RX pin is used as a UART receiver pin.

At this point the receiver will be enabled which will begin to look for a start bit.

When a character is received, the following sequence of events will occur:

- The RXIF bit in the USR register will be set then RXR register has data available, at least one character can be read.
- When the contents of the shift register have been transferred to the RXR register and if the RIE bit is set, then an interrupt will be generated.
- If during reception, a frame error, noise error, parity error or an overrun error has been detected, and then the error flags can be set.

The RXIF bit can be cleared using the following software sequence:

- A USR register access
- A RXR register read execution

Receiving Break

Any break character received by the UART will be managed as a framing error. The receiver will count and expect a certain number of bit times as specified by the values programmed into the BNO and STOPS bits. If the break is much longer than 13 bit times, the reception will be considered as complete after the number of bit times specified by BNO and STOPS. The RXIF bit is set, FERR is set, zeros are loaded into the receive data register, interrupts are generated if appropriate and the RIDLE bit is set. If a long break signal has been detected and the receiver has received a start bit, the data bits and the invalid stop bit, which sets the FERR flag, the receiver must wait for a valid stop bit before looking for the next start bit. The receiver will not make the assumption that the break condition on the line is the next start bit. A break is regarded as a character that contains only zeros with the FERR flag set. The break character will be loaded into the buffer and no further data will be received until stop bits are received. It should be noted that the RIDLE read only flag will go high when the stop bits have not yet been received. The reception of a break character on the UART registers will result in the following:

- The framing error flag, FERR, will be set.
- The receive data register, RXR, will be cleared.
- The OERR, NF, PERR, RIDLE or RXIF flags will possibly be set.

Idle Status

When the receiver is reading data, which means it will be in between the detection of a start bit and the reading of a stop bit, the receiver status flag in the USR register, otherwise known as the RIDLE flag, will have a zero value. In between the reception of a stop bit and the detection of the next start bit, the RIDLE flag will have a high value, which indicates the receiver is in an idle condition.

Receiver Interrupt

The read only receive interrupt flag RXIF in the USR register is set by an edge generated by the receiver. An interrupt is generated if RIE=1, when a word is transferred from the Receive Shift Register, RSR, to the Receive Data Register, RXR. An overrun error can also generate an interrupt if RIE=1.

Managing Receiver Errors

Several types of reception errors can occur within the UART module, the following section describes the various types and how they are managed by the UART.

Overrun Error – OERR

The RXR register is composed of a two byte deep FIFO data buffer, where two bytes can be held in the FIFO register, while a third byte can continue to be received. Before the third byte has been entirely shifted in, the data should be read from the RXR register. If this is not done, the overrun error flag OERR will be consequently indicated.

In the event of an overrun error occurring, the following will happen:

- The OERR flag in the USR register will be set.
- The RXR contents will not be lost.
- The shift register will be overwritten.
- An interrupt will be generated if the RIE bit is set.

The OERR flag can be cleared by an access to the USR register followed by a read to the RXR register.

Noise Error – NF

Over-sampling is used for data recovery to identify valid incoming data and noise. If noise is detected within a frame, the following will occur:

- The read only noise flag, NF, in the USR register will be set on the rising edge of the RXIF bit.
- Data will be transferred from the shift register to the RXR register.
- No interrupt will be generated. However this bit rises at the same time as the RXIF bit which itself generates an interrupt.

Note that the NF flag is reset by a USR register read operation followed by an RXR register read operation.

Framing Error – FERR

The read only framing error flag, FERR, in the USR register, is set if a zero is detected instead of stop bits. If two stop bits are selected, only the first stop bit is detected and it must be high. If the first stop bit is low, the FERR flag will be set. The FERR flag is buffered along with the received data and is cleared in any reset.

Parity Error – PERR

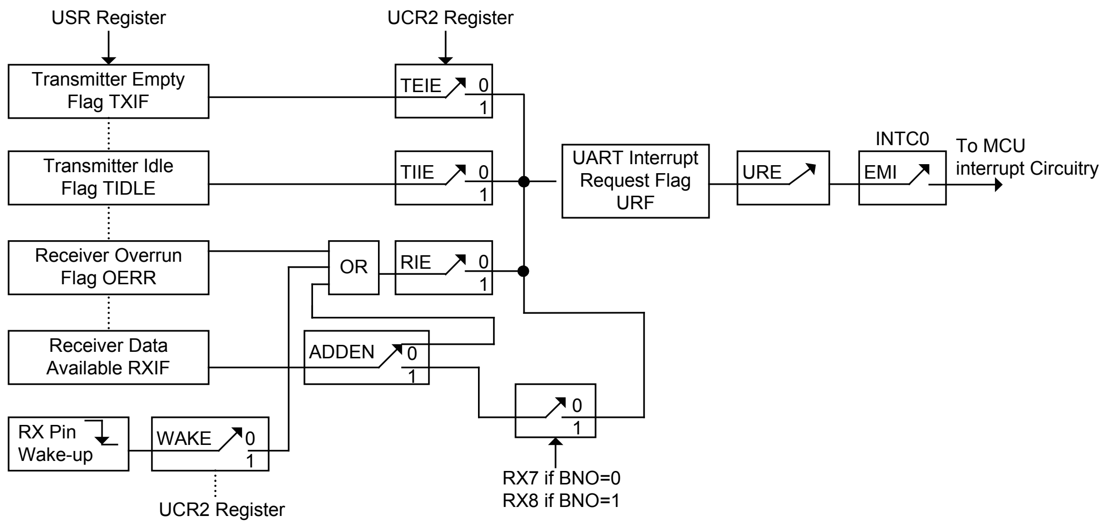
The read only parity error flag, PERR, in the USR register, is set if the parity of the received word is incorrect. This error flag is only applicable if the parity function is enabled, PREN=1, and if the parity type, odd or even, is selected. The read only PERR flag is buffered along with the received data bytes. It is cleared on any reset, it should be noted that the FERR and PERR flags are buffered along with the corresponding word and should be read before reading the data word.

UART Interrupt Structure

Several individual UART conditions can generate a UART interrupt. When these conditions exist, a low pulse will be generated to get the attention of the microcontroller. These conditions are a transmitter data register empty, transmitter idle, receiver data available, receiver overrun, address detect and an RX pin wake-up. When any of these conditions are created, if its corresponding interrupt control is enabled and the stack is not full, the program will jump to its corresponding interrupt vector where it can be serviced before returning to the main program. Four of these conditions have the corresponding USR register flags which will generate a UART interrupt if its associated interrupt enable control bit in the UCR2 register is set. The two transmitter interrupt conditions have their own corresponding enable control bits, while the two receiver interrupt conditions have a shared enable control bit. These enable bits can be used to mask out individual UART interrupt sources.

The address detect condition, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt when an address detect condition occurs if its function is enabled by setting the ADDEN bit in the UCR2 register. An RX pin wake-up, which is also a UART interrupt source, does not have an associated flag, but will generate a UART interrupt if the microcontroller is woken up by a falling edge on the RX pin, if the WAKE and RIE bits in the UCR2 register are set. Note that in the event of an RX wake-up interrupt occurring, there will be a certain period of delay, commonly known as the System Start-up Time, for the oscillator to restart and stabilize before the system resumes normal operation.

Note that the USR register flags are read only and cannot be cleared or set by the application program, neither will they be cleared when the program jumps to the corresponding interrupt servicing routine, as is the case for some of the other interrupts. The flags will be cleared automatically when certain actions are taken by the UART, the details of which are given in the UART register section. The overall UART interrupt can be disabled or enabled by the related interrupt enable control bits in the interrupt control registers of the microcontroller to decide whether the interrupt requested by the UART module is masked out or allowed.


USR Register
UCR2 Register
Transmitter Empty Flag TXIF
Transmitter Idle Flag TIDLE
Receiver Overrun Flag OERR
Receiver Data Available RXIF
RX Pin Wake-up
WAKE
UCR2 Register
TEIE
TIIE
RIE
OR
ADDEN
RX7 if BNO=0
RX8 if BNO=1
UART Interrupt Request Flag URF
URE
INTC0
EMI
To MCU interrupt Circuitry

UART Interrupt Structure

Address Detect Mode

Setting the Address Detect function enable control bit, ADDEN, in the UCR2 register, enables this special function. If this bit is set to 1, then an additional qualifier will be placed on the generation of a Receiver Data Available interrupt, which is requested by the RXIF flag. If the ADDEN bit is equal to 1, then when the data is available, an interrupt will only be generated, if the highest received bit has a high value. Note that the related interrupt enable control bit and the EMI bit of the microcontroller must also be enabled for correct interrupt generation. The highest address bit is the 9th bit if the bit BNO=1 or the 8th bit if the bit BNO=0. If the highest bit is high, then the received word will be defined as an address rather than data. A Data Available interrupt will be generated every time the last bit of the received word is set. If the ADDEN bit is equal to 0, then a Receive Data Available interrupt will be generated each time the RXIF flag is set, irrespective of the data last but status. The address detection and parity functions are mutually exclusive functions. Therefore, if the address detect function is enabled, then to ensure correct operation, the parity function should be disabled by resetting the parity function enable bit PREN to zero.

ADDENn	Bit 9 if BNO=1 Bit 8 if BNO=0	UART Interrupt Generated
0	0	v
	1	v
1	0	x
	1	v

ADDEN Bit Function

UART Wake-up

When the MCU enters the Power Down Mode and the system clock is stopped, the UART interface will cease to function since the UART driven clock source is derived from the system clock. If the MCU enters the Power Down Mode while a transmission is still in progress, then the transmission will be paused until the UART clock source derived from the microcontroller is activated. In a similar way, if the MCU enters the Power Down Mode while receiving data, then the reception of data will likewise be paused. When the MCU enters the Power Down Mode, note that the USR, UCR1, UCR2, transmit and receive registers, as well as the BRG register will not be affected. It is recommended to make sure first that the UART data transmission or reception has been finished before the microcontroller enters the Power Down mode.

The UART function contains a receiver RX pin wake-up function, which is enabled or disabled by the WAKE bit in the UCR2 register. If this bit, along with the UART enable bit, UARTEN, the receiver enable bit, RXEN and the receiver interrupt bit, RIE, are all set before the MCU enters the Power Down Mode, then a falling edge on the RX pin will wake up the MCU from the Power Down Mode. Note that as it takes certain system clock cycles after a wake-up, before normal microcontroller operation resumes, any data received during this time on the RX pin will be ignored.

For a UART wake-up interrupt to occur, in addition to the bits for the wake-up being set, the global interrupt enable bit, EMI, and the UART interrupt enable bit, URE, must also be set. If these two bits are not set then only a wake up event will occur and no interrupt will be generated. Note also that as it takes certain system clock cycles after a wake-up before normal microcontroller resumes, the UART interrupt will not be generated until after this time has elapsed.

Interrupts

Interrupts are an important part of any microcontroller system. When an external event or an internal function such as a Timer Module requires microcontroller attention, their corresponding interrupt will enforce a temporary suspension of the main program allowing the microcontroller to direct attention to their respective needs. The device contains several external interrupt and internal interrupts functions. The external interrupts are generated by the action of the external INT0~INT1 pins, while the internal interrupts are generated by various internal functions such as the TMs, Time Base and LVD.

Interrupt Registers

Overall interrupt control, which basically means the setting of request flags when certain microcontroller conditions occur and the setting of interrupt enable bits by the application program, is controlled by a series of registers, located in the Special Purpose Data Memory, as shown in the accompanying table. The number of registers depends upon the device chosen but fall into three categories. The first is the INTC0~INTC2 registers which setup the primary interrupts, the second is the MFI0~MFI2 registers which setup the Multi-function interrupts. Finally there is an INTEG register to setup the external interrupt trigger edge type.

Each register contains a number of enable bits to enable or disable individual registers as well as interrupt flags to indicate the presence of an interrupt request. The naming convention of these follows a specific pattern. First is listed an abbreviated interrupt type, then the (optional) number of that interrupt followed by either an "E" for enable/disable bit or "F" for request flag.

Function	Enable Bit	Request Flag	Notes
Global	EMI	—	—
INTn Pin	INTnE	INTnF	n=0~1
Time Base	TBnE	TBnF	n=0~1
Multi-function	MFnE	MFnF	n=0~2
I^{2}C Interface	IICE	IICF	—
UART Interface	URE	URF	—
LVD	LVE	LVF	—
EEPROM	DEE	DEF	—
CTM	CTMPE	CTMPF	—
	CTMAE	CTMAF	—
ETM	ETMPE	ETMPF	—
	ETMAE	ETMAF	—
	ETMBE	ETMBF	—
TM	PTMPE	PTMPF	—
	PTMAE	PTMAF	—

Interrupt Register Bit Naming Conventions

Name	Bit							
	7	6	5	4	3	2	1	0
INTEG	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
INTC0	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
INTC1	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
INTC2	—	—	URF	IICF	—	—	URE	IICE
MFI0	PTMAF	PTMPF	CTMAF	CTMPF	PTMAE	PTMPE	CTMAE	CTMPE
MFI1	—	ETMBF	ETMAF	ETMPF	—	ETMBE	ETMAE	ETMPE
MFI2	—	—	DEF	LVF	—	—	DEE	LVE

Interrupt Registers List

INTEG Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	—	—	INT1S1	INT1S0	INT0S1	INT0S0
R/W	—	—	—	—	R/W	R/W	R/W	R/W
POR	—	—	—	—	0	0	0	0

Bit 7~4 Unimplemented, read as "0"

Bit 3~2 **INT1S1, INT1S0:** interrupt edge control for INT1 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edges

Bit 1~0 **INT0S1, INT0S0:** interrupt edge control for INT0 pin
00: Disable
01: Rising edge
10: Falling edge
11: Rising and falling edge

INTC0 Register

Bit	7	6	5	4	3	2	1	0
Name	—	TB0F	INT1F	INT0F	TB0E	INT1E	INT0E	EMI
R/W	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	—	0	0	0	0	0	0	0

Bit 7 Unimplemented, read as "0"

Bit 6 **TB0F:** Time Base 0 interrupt request flag
0: No request
1: Interrupt request

Bit 5 **INT1F:** INT1 interrupt request flag
0: No request
1: Interrupt request

Bit 4 **INT0F:** INT0 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **TB0E:** Time Base 0 interrupt control
0: Disable
1: Enable

Bit 2 **INT1E:** INT1 interrupt control
0: Disable
1: Enable

Bit 1 **INT0E:** INT0 interrupt control
0: Disable
1: Enable

Bit 0 **EMI:** Global interrupt control
0: Disable
1: Enable

INTC1 Register

Bit	7	6	5	4	3	2	1	0
Name	MF2F	MF1F	MF0F	TB1F	MF2E	MF1E	MF0E	TB1E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **MF2F:** Multi-function interrupt 2 request flag
0: No request
1: Interrupt request

Bit 6 **MF1F:** Multi-function interrupt 1 request flag
0: No request
1: Interrupt request

Bit 5 **MF0F:** Multi-function interrupt 0 request flag
0: No request
1: Interrupt request

Bit 4 **TB1F:** Time Base 1 interrupt request flag
0: No request
1: Interrupt request

Bit 3 **MF2E:** Multi-function interrupt 2 control
0: Disable
1: Enable

Bit 2 **MF1E:** Multi-function interrupt 1 control
0: Disable
1: Enable

Bit 1 **MF0E:** Multi-function interrupt 0 control
0: Disable
1: Enable

Bit 0 **TB1E:** Time Base 1 interrupt control
0: Disable
1: Enable

INTC2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	URF	IICF	—	—	URE	IICE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6 unimplemented, read as "0"

Bit 5 **URF:** UART interface interrupt request flag
0: No request
1: Interrupt request

Bit 4 **IICF:** I^2C interface interrupt request flag
0: No request
1: Interrupt request

Bit 3~2 unimplemented, read as "0"

Bit 1 **URE:** UART interface interrupt control
0: Disable
1: Enable

Bit 0 **IICE:** I^2C interface interrupt control
0: Disable
1: Enable

MFI0 Register

Bit	7	6	5	4	3	2	1	0
Name	PTMAF	PTMPF	CTMAF	CTMPF	PTMAE	PTMPE	CTMAE	CTMPE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	0	0	0	0	0	0

Bit 7 **PTMAF:** PTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 6 **PTMPF:** PTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **CTMAF:** CTM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **CTMPF:** CTM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 **PTMAE:** PTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 2 **PTMPE:** PTM Comparator P match interrupt control
0: Disable
1: Enable

Bit 1 **CTMAE:** CTM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **CTMPE:** CTM Comparator P match interrupt control
0: Disable
1: Enable

MFI1 Register

Bit	7	6	5	4	3	2	1	0
Name	—	ETMBF	ETMAF	ETMPF	—	ETMBE	ETMAE	ETMPE
R/W	—	R/W	R/W	R/W	—	R/W	R/W	R/W
POR	—	0	0	0	—	0	0	0

Bit 7 unimplemented, read as "0"

Bit 6 **ETMBF:** ETM Comparator B match interrupt request flag
0: No request
1: Interrupt request

Bit 5 **ETMAF:** ETM Comparator A match interrupt request flag
0: No request
1: Interrupt request

Bit 4 **ETMPF:** ETM Comparator P match interrupt request flag
0: No request
1: Interrupt request

Bit 3 unimplemented, read as "0"

Bit 2 **ETMBE:** ETM Comparator B match interrupt control
0: Disable
1: Enable

Bit 1 **ETMAE:** ETM Comparator A match interrupt control
0: Disable
1: Enable

Bit 0 **ETMPE:** ETM Comparator P match interrupt control
0: Disable
1: Enable

MFI2 Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	DEF	LVF	—	—	DEE	LVE
R/W	—	—	R/W	R/W	—	—	R/W	R/W
POR	—	—	0	0	—	—	0	0

Bit 7~6	Unimplemented, read as "0"
Bit 5	**DEF:** Data EEPROM interrupt request flag 0: No request 1: Interrupt request
Bit 4	**LVF:** LVD interrupt request flag 0: No request 1: Interrupt request
Bit 3~2	Unimplemented, read as "0"
Bit 1	**DEE:** Data EEPROM Interrupt Control 0: Disable 1: Enable
Bit 0	**LVE:** LVD Interrupt Control 0: Disable 1: Enable

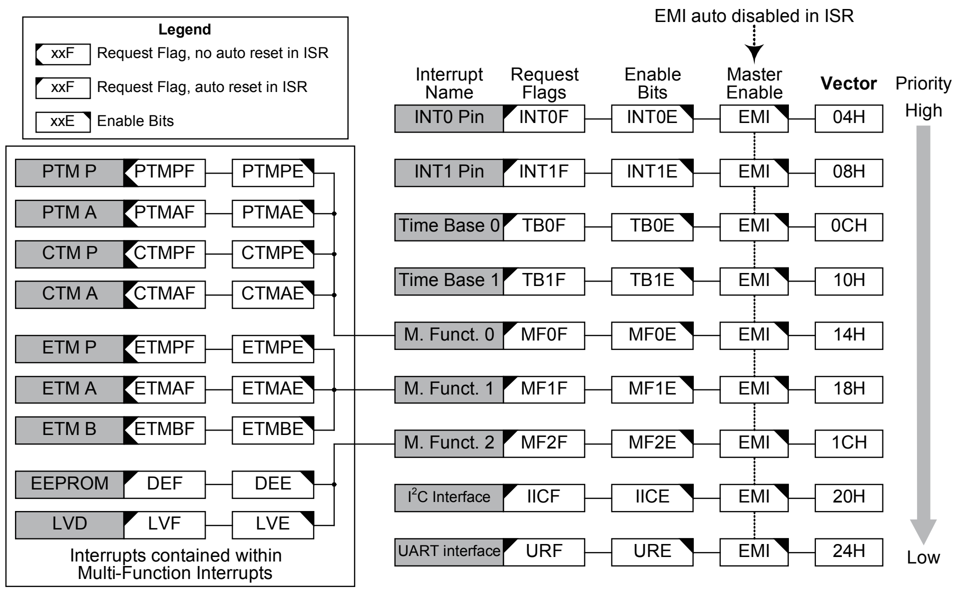
Interrupt Operation

When the conditions for an interrupt event occur, such as a TM Comparator P, Comparator A or Comparator B match etc, the relevant interrupt request flag will be set. Whether the request flag actually generates a program jump to the relevant interrupt vector is determined by the condition of the interrupt enable bit. If the enable bit is set high then the program will jump to its relevant vector; if the enable bit is zero then although the interrupt request flag is set an actual interrupt will not be generated and the program will not jump to the relevant interrupt vector. The global interrupt enable bit, if cleared to zero, will disable all interrupts.

When an interrupt is generated, the Program Counter, which stores the address of the next instruction to be executed, will be transferred onto the stack. The Program Counter will then be loaded with a new address which will be the value of the corresponding interrupt vector. The microcontroller will then fetch its next instruction from this interrupt vector. The instruction at this vector will usually be a "JMP" which will jump to another section of program which is known as the interrupt service routine. Here is located the code to control the appropriate interrupt. The interrupt service routine must be terminated with a "RETI" , which retrieves the original Program Counter address from the stack and allows the microcontroller to continue with normal execution at the point where the interrupt occurred.

The various interrupt enable bits, together with their associated request flags, are shown in the accompanying diagrams with their order of priority. Some interrupt sources have their own individual vector while others share the same multi-function interrupt vector. Once an interrupt subroutine is serviced, all the other interrupts will be blocked, as the global interrupt enable bit, EMI bit will be cleared automatically. This will prevent any further interrupt nesting from occurring. However, if other interrupt requests occur during this interval, although the interrupt will not be immediately serviced, the request flag will still be recorded.

If an interrupt requires immediate servicing while the program is already in another interrupt service routine, the EMI bit should be set after entering the routine, to allow interrupt nesting. If the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the Stack Pointer is decremented. If immediate service is desired, the stack must be prevented from becoming full. In case of simultaneous requests, the accompanying diagram shows the priority that is applied. All of the interrupt request flags when set will wake-up the device if it is in SLEEP or IDLE Mode, however to prevent a wake-up from occurring the corresponding flag should be set before the device is in SLEEP or IDLE Mode.


Legend
xxF Request Flag, no auto reset in ISR
xxF Request Flag, auto reset in ISR
xxE Enable Bits
EMI auto disabled in ISR
Interrupt Name
Request Flags
Enable Bits
Master Enable
Vector
Priority
High
Low
PTM P PTMPF PTMPE
PTM A PTMAF PTMAE
CTM P CTMPF CTMPE
CTM A CTMAF CTMAE
ETM P ETMPF ETMPE
ETM A ETMAF ETMAE
ETM B ETMBF ETMBE
EEPROM DEF DEE
LVD LVF LVE
Interrupts contained within Multi-Function Interrupts
INT0 Pin INT0F INT0E EMI 04H
INT1 Pin INT1F INT1E EMI 08H
Time Base 0 TB0F TB0E EMI 0CH
Time Base 1 TB1F TB1E EMI 10H
M. Funct. 0 MF0F MF0E EMI 14H
M. Funct. 1 MF1F MF1E EMI 18H
M. Funct. 2 MF2F MF2E EMI 1CH
I2C Interface IICF IICE EMI 20H
UART interface URF URE EMI 24H

Interrupt Structure

External Interrupt

The external interrupts are controlled by signal transitions on the pins INT0~INT1. An external interrupt request will take place when the external interrupt request flags, INT0F~INT1F, are set, which will occur when a transition, whose type is chosen by the edge select bits, appears on the external interrupt pins. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and respective external interrupt enable bit, INT0E~INT1E, must first be set. Additionally the correct interrupt edge type must be selected using the INTEG register to enable the external interrupt function and to choose the trigger edge type. As the external interrupt pins are pin-shared with I/O pins, they can only be configured as external interrupt pins if their external interrupt enable bit in the corresponding interrupt register has been set. The pin must also be setup as an input by setting the corresponding bit in the port control register. When the interrupt is enabled, the stack is not full and the correct transition type appears on the external interrupt pin, a subroutine call to the external interrupt vector, will take place. When the interrupt is serviced, the external interrupt request flags, INT0F~INT1F, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts. Note that any pull-high resistor selections on the external interrupt pins will remain valid even if the pin is used as an external interrupt input.

The INTEG register is used to select the type of active edge that will trigger the external interrupt. A choice of either rising or falling or both edge types can be chosen to trigger an external interrupt. Note that the INTEG register can also be used to disable the external interrupt function.

Multi-function Interrupt

Within this device there are up to three Multi-function interrupts. Unlike the other independent interrupts, these interrupts have no independent source, but rather are formed from other existing interrupt sources, namely the TM Interrupts, EEPROM Interrupt and LVD interrupt.

A Multi-function interrupt request will take place when any of the Multi-function interrupt request flags, MFnF, are set. The Multi-function interrupt flags will be set when any of their included functions generate an interrupt request flag. To allow the program to branch to its respective interrupt vector address, when the Multi-function interrupt is enabled and the stack is not full, and either one of the interrupts contained within each of Multi-function interrupt occurs, a subroutine call to one of the Multi-function interrupt vectors will take place. When the interrupt is serviced, the related Multi-Function request flag, will be automatically reset and the EMI bit will be automatically cleared to disable other interrupts.

However, it must be noted that, although the Multi-function Interrupt flags will be automatically reset when the interrupt is serviced, the request flags from the original source of the Multi-function interrupts, namely the TM Interrupts, EEPROM Interrupt and LVD interrupt will not be automatically reset and must be manually reset by the application program.

Time Base Interrupt

The function of the Time Base Interrupt is to provide regular time signal in the form of an internal interrupt. It is controlled by the overflow signal from its internal timer. When this happens its interrupt request flag, TBnF, will be set. To allow the program to branch to its respective interrupt vector addresses, the global interrupt enable bit, EMI and Time Base enable bit, TBnE, must first be set. When the interrupt is enabled, the stack is not full and the Time Base overflows, a subroutine call to its respective vector location will take place. When the interrupt is serviced, the interrupt request flag, TBnF, will be automatically reset and the EMI bit will be cleared to disable other interrupts.

The purpose of the Time Base Interrupt is to provide an interrupt signal at fixed time periods. Its clock source, f_{TB}, originates from the internal clock source f_{SUB} or $f_{SYS}/4$. And then passes through a divider, the division ratio of which is selected by programming the appropriate bits in the TBC register to obtain longer interrupt periods whose value ranges. The clock source which in turn controls the Time Base interrupt period is selected using a bit in the TBC register.

TBC Register

Bit	7	6	5	4	3	2	1	0
Name	TBON	TBCK	TB11	TB10	LXTLP	TB02	TB01	TB00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
POR	0	0	1	1	0	1	1	1

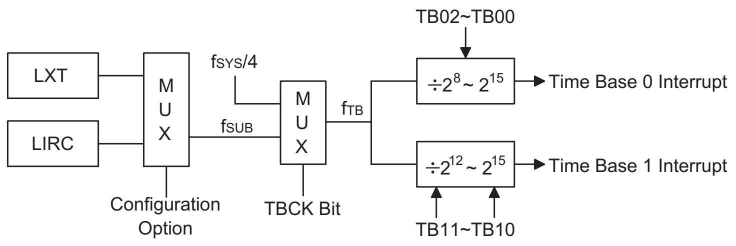
Bit 7 **TBON:** Time Base 0 and Time Base 1 Enable/Disable
0: Disable
1: Enable

Bit 6 **TBCK:** TB Clock f_{TB} Select
0: f_{SUB}
1: $f_{SYS}/4$

Bit 5~4 **TB11~TB10:** Time Base 1 Time-out Period Selection
00: $2^{12}/f_{TB}$
01: $2^{13}/f_{TB}$
10: $2^{14}/f_{TB}$
11: $2^{15}/f_{TB}$

Bit 3 **LXTLP:** LXT Low Power Control
0: Disable (LXT quick start-up)
1: Enable (LXT low power start-up)

Bit 2~0 **TB02~TB00:** Time Base 0 Time-out Period
000: $2^8/f_{TB}$
001: $2^9/f_{TB}$
010: $2^{10}/f_{TB}$
011: $2^{11}/f_{TB}$
100: $2^{12}/f_{TB}$
101: $2^{13}/f_{TB}$
110: $2^{14}/f_{TB}$
111: $2^{15}/f_{TB}$



Time Base Interrupt

I^{2}C Interrupt

An I^{2}C Interrupt request will take place when the I^{2}C Interrupt request flag, IICF, is set when the I^{2}C time-out occurs, slave address matches or data byte transfer completes. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the I^{2}C Interrupt enable bit, IICE, must first be set. When the interrupt is enabled, the stack is not full and one of the I^{2}C interrupt event occurs, a subroutine call to the respective Interrupt vector, will take place. When the Interrupt is serviced, the interrupt request flag will be automatically cleared and the EMI bit will be automatically cleared to disable other interrupts.

UART Interrupt

An UART Interrupt request will take place when the UART Interrupt request flag, URF, is set when any one of the UART interrupt events occurs, which are described in the "UART Interrupt Structure" section. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the UART Interrupt enable bit, URE, must first be set. When the interrupt is enabled, the stack is not full and one of the UART interrupt event occurs, a subroutine call to the respective Interrupt vector, will take place. When the Interrupt is serviced, the interrupt request flag will be automatically cleared and the EMI bit will be automatically cleared to disable other interrupts.

EEPROM Interrupt

The EEPROM interrupt is contained within the Multi-function Interrupt. An EEPROM Interrupt request will take place when the EEPROM Interrupt request flag, DEF, is set, which occurs when an EEPROM write operation ends. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, and the EEPROM Interrupt enable bit, DEE, and Muti-function interrupt enable bits, must first be set. When the interrupt is enabled, the stack is not full and an EEPROM write operation ends, a subroutine call to the respective Multi-function Interrupt vector, will take place.When the EEPROM Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the DEF flag will not be automatically cleared, it has to be cleared by the application program.

LVD Interrupt

The Low Voltage Detector Interrupt is contained within the Multi-function Interrupt. An LVD Interrupt request will take place when the LVD Interrupt request flag, LVF, is set, which occurs when the Low Voltage Detector function detects a low power supply voltage. To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, Low Voltage Interrupt enable bit, LVE, and associated Multi-function interrupt enable bit, must first be set. When the interrupt is enabled, the stack is not full and a low voltage condition occurs, a subroutine call to the Multi-function Interrupt vector, will take place. When the Low Voltage Interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the Multi-function interrupt request flag will be also automatically cleared. As the LVF flag will not be automatically cleared, it has to be cleared by the application program.

TM Interrupts

The Compact and Periodic Type TMs have two interrupts each, while the Enhanced Type TM has three interrupts. All of the TM interrupts are contained within the Multi-function Interrupts. For each of the Compact and Periodic Type TMs there are two interrupt request flags, xTMPF and xTMAF (x=C or P), and two enable bits xTMPE and xTMAE (x=C or P). For the Enhanced Type TM there are three interrupt request flags ETMPF, ETMAF and ETMBF and three enable bits ETMPE, ETMAE and ETMBE. A TM interrupt request will take place when any of the TM request flags are set, a situation which occurs when a TM comparator P, A or B match situation happens.

To allow the program to branch to its respective interrupt vector address, the global interrupt enable bit, EMI, respective TM Interrupt enable bit, and relevant Multi-function Interrupt enable bit, MFnE, must first be set. When the interrupt is enabled, the stack is not full and a TM comparator match situation occurs, a subroutine call to the relevant Multi-function Interrupt vector locations, will take place. When the TM interrupt is serviced, the EMI bit will be automatically cleared to disable other interrupts, however only the related MFnF flag will be automatically cleared. As the TM interrupt request flags will not be automatically cleared, they have to be cleared by the application program.

Interrupt Wake-up Function

Each of the interrupt functions has the capability of waking up the microcontroller when in the SLEEP or IDLE Mode. A wake-up is generated when an interrupt request flag changes from low to high and is independent of whether the interrupt is enabled or not. Therefore, even though the device is in the SLEEP or IDLE Mode and its system oscillator stopped, situations such as external edge transitions on the external interrupt pins, a low power supply voltage or comparator input change may cause their respective interrupt flag to be set high and consequently generate an interrupt. Care must therefore be taken if spurious wake-up situations are to be avoided. If an interrupt wake-up function is to be disabled then the corresponding interrupt request flag should be set high before the device enters the SLEEP or IDLE Mode. The interrupt enable bits have no effect on the interrupt wake-up function.

Programming Considerations

By disabling the relevant interrupt enable bits, a requested interrupt can be prevented from being serviced, however, once an interrupt request flag is set, it will remain in this condition in the interrupt register until the corresponding interrupt is serviced or until the request flag is cleared by the application program.

Where a certain interrupt is contained within a Multi-function interrupt, then when the interrupt service routine is executed, as only the Multi-function interrupt request flags, MF0F~MF2F, will be automatically cleared, the individual request flag for the function needs to be cleared by the application program.

It is recommended that programs do not use the "CALL" instruction within the interrupt service subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately. If only one stack is left and the interrupt is not well controlled, the original control sequence will be damaged once a CALL subroutine is executed in the interrupt subroutine.

Every interrupt has the capability of waking up the microcontroller when it is in SLEEP or IDLE Mode, the wake up being generated when the interrupt request flag changes from low to high. If it is required to prevent a certain interrupt from waking up the microcontroller then its respective request flag should be first set high before enter SLEEP or IDLE Mode.

As only the Program Counter is pushed onto the stack, then when the interrupt is serviced, if the contents of the accumulator, status register or other registers are altered by the interrupt service program, their contents should be saved to the memory at the beginning of the interrupt service routine.

To return from an interrupt subroutine, either a RET or RETI instruction may be executed. The RETI instruction in addition to executing a return to the main program also automatically sets the EMI bit high to allow further interrupts. The RET instruction however only executes a return to the main program leaving the EMI bit in its present zero state and therefore disabling the execution of further interrupts.

Low Voltage Detector – LVD

The device has a Low Voltage Detector function, also known as LVD. This enabled the device to monitor the power supply voltage, V_{DD}, and provide a warning signal should it fall below a certain level. This function may be especially useful in battery applications where the supply voltage will gradually reduce as the battery ages, as it allows an early warning battery low signal to be generated. The Low Voltage Detector also has the capability of generating an interrupt signal.

LVD Register

The Low Voltage Detector function is controlled using a single register with the name LVDC. Three bits in this register, VLVD2~VLVD0, are used to select one of eight fixed voltages below which a low voltage condition will be detemined. A low voltage condition is indicated when the LVDO bit is set. If the LVDO bit is low, this indicates that the V_{DD} voltage is above the preset low voltage value. The LVDEN bit is used to control the overall on/off function of the low voltage detector. Setting the bit high will enable the low voltage detector. Clearing the bit to zero will switch off the internal low voltage detector circuits. As the low voltage detector will consume a certain amount of power, it may be desirable to switch off the circuit when not in use, an important consideration in power sensitive battery powered applications.

LVDC Register

Bit	7	6	5	4	3	2	1	0
Name	—	—	LVDO	LVDEN	—	VLVD2	VLVD1	VLVD0
R/W	—	—	R	R/W	—	R/W	R/W	R/W
POR	—	—	0	0	—	0	0	0

Bit 7~6 Unimplemented, read as "0"

Bit 5 **LVDO:** LVD Output Flag
0: No Low Voltage Detected
1: Low Voltage Detected

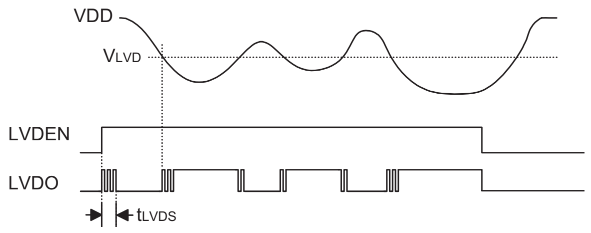
Bit 4 **LVDEN:** Low Voltage Detector Enable/Disable
0: Disable
1: Enable

Bit 3 Unimplemented, read as "0"

Bit 2~0 **VLVD2~VLVD0:** Select LVD Voltage
000: 2.0V
001: 2.2V
010: 2.4V
011: 2.7V
100: 3.0V
101: 3.3V
110: 3.6V
111: 4.0V

LVD Operation

The Low Voltage Detector function operates by comparing the power supply voltage, V_{DD}, with a pre-specified voltage level stored in the LVDC register. This has a range of between 2.0V and 4.0V. When the power supply voltage, V_{DD}, falls below this pre-determined value, the LVDO bit will be set high indicating a low power supply voltage condition. The Low Voltage Detector function is supplied by a reference voltage which will be automatically enabled. When the device is powered down the low voltage detector will remain active if the LVDEN bit is high. After enabling the Low Voltage Detector, a time delay t_{LVDS} should be allowed for the circuitry to stabilise before reading the LVDO bit. Note also that as the V_{DD} voltage may rise and fall rather slowly, at the voltage nears that of V_{LVD}, there may be multiple bit LVDO transitions.



LVD Operation

The Low Voltage Detector also has its own interrupt which is contained within one of the Multi-function interrupts, providing an alternative means of low voltage detection, in addition to polling the LVDO bit. The interrupt will only be generated after a delay of t_{LVD} after the LVDO bit has been set high by a low voltage condition. When the device is powered down the Low Voltage Detector will remain active if the LVDEN bit is high. In this case, the LVF interrupt request flag will be set, causing an interrupt to be generated if V_{DD} falls below the preset LVD voltage. This will cause the device to wake-up from the SLEEP or IDLE Mode, however if the Low Voltage Detector wake up function is not required then the LVF flag should be first set high before the device enters the SLEEP or IDLE Mode.

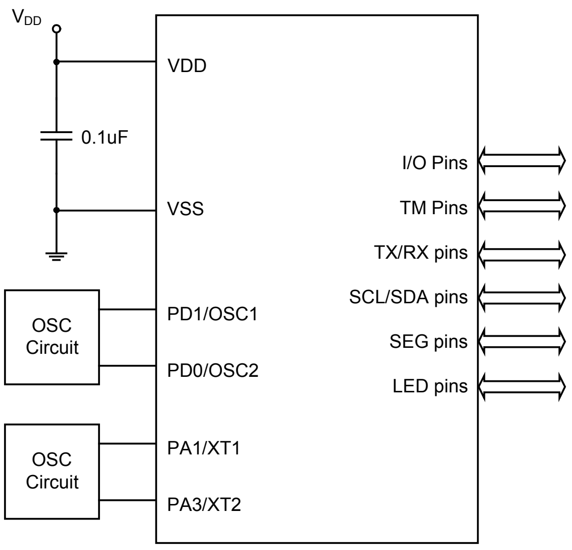
When LVD function is enabled, it is recommended to clear LVD flag first, and then enables interrupt function to avoid mistake action.

Configuration Options

Configuration options refer to certain options within the MCU that are programmed into the device during the programming process. During the development process, these options are selected using the HT-IDE software development tools. As these options are programmed into the device using the hardware programming tools, once they are selected they cannot be changed later using the application program. All options must be defined for proper system function, the details of which are shown in the table.

No.	Options
1	High Speed System Oscillator Selection – f_H HXT or HIRC
2	Low Speed System Oscillator Selection – f_{SUB} LXT or LIRC
3	HIRC Frequency Selection 4MHz, 8MHz or 12MHz

Application Circuits



Instruction Set

Introduction

Central to the successful operation of any microcontroller is its instruction set, which is a set of program instruction codes that directs the microcontroller to perform certain operations. In the case of Holtek microcontrollers, a comprehensive and flexible set of over 60 instructions is provided to enable programmers to implement their application with the minimum of programming overheads.

For easier understanding of the various instruction codes, they have been subdivided into several functional groupings.

Instruction Timing

Most instructions are implemented within one instruction cycle. The exceptions to this are branch, call, or table read instructions where two instruction cycles are required. One instruction cycle is equal to 4 system clock cycles, therefore in the case of an 8MHz system oscillator, most instructions would be implemented within 0.5µs and branch or call instructions would be implemented within 1µs. Although instructions which require one more cycle to implement are generally limited to the JMP, CALL, RET, RETI and table read instructions, it is important to realize that any other instructions which involve manipulation of the Program Counter Low register or PCL will also take one more cycle to implement. As instructions which change the contents of the PCL will imply a direct jump to that new address, one more cycle will be required. Examples of such instructions would be ″CLR PCL″ or ″MOV PCL, A″. For the case of skip instructions, it must be noted that if the result of the comparison involves a skip operation then this will also take one more cycle, if no skip is involved then only one cycle is required.

Moving and Transferring Data

The transfer of data within the microcontroller program is one of the most frequently used operations. Making use of three kinds of MOV instructions, data can be transferred from registers to the Accumulator and vice-versa as well as being able to move specific immediate data directly into the Accumulator. One of the most important data transfer applications is to receive data from the input ports and transfer data to the output ports.

Arithmetic Operations

The ability to perform certain arithmetic operations and data manipulation is a necessary feature of most microcontroller applications. Within the Holtek microcontroller instruction set are a range of add and subtract instruction mnemonics to enable the necessary arithmetic to be carried out. Care must be taken to ensure correct handling of carry and borrow data when results exceed 255 for addition and less than 0 for subtraction. The increment and decrement instructions INC, INCA, DEC and DECA provide a simple means of increasing or decreasing by a value of one of the values in the destination specified.

Logical and Rotate Operations

The Periodic logical operations such as AND, OR, XOR and CPL all have their own instruction within the Holtek microcontroller instruction set. As with the case of most instructions involving data manipulation, data must pass through the Accumulator which may involve additional programming steps. In all logical data operations, the zero flag may be set if the result of the operation is zero. Another form of logical data manipulation comes from the rotate instructions such as RR, RL, RRC and RLC which provide a simple means of rotating one bit right or left. Different rotate instructions exist depending on program requirements. Rotate instructions are useful for serial port programming applications where data can be rotated from an internal register into the Carry bit from where it can be examined and the necessary serial bit set high or low. Another application where rotate data operations are used is to implement multiplication and division calculations.

Branches and Control Transfer

Program branching takes the form of either jumps to specified locations using the JMP instruction or to a subroutine using the CALL instruction. They differ in the sense that in the case of a subroutine call, the program must return to the instruction immediately when the subroutine has been carried out. This is done by placing a return instruction RET in the subroutine which will cause the program to jump back to the address right after the CALL instruction. In the case of a JMP instruction, the program simply jumps to the desired location. There is no requirement to jump back to the original jumping off point as in the case of the CALL instruction. One special and extremely useful set of branch instructions are the conditional branches. Here a decision is first made regarding the condition of a certain data memory or individual bits. Depending upon the conditions, the program will continue with the next instruction or skip over it and jump to the following instruction. These instructions are the key to decision making and branching within the program perhaps determined by the condition of certain input switches or by the condition of internal data bits.

Bit Operations

The ability to provide single bit operations on Data Memory is an extremely flexible feature of all Holtek microcontrollers. This feature is especially useful for output port bit programming where individual bits or port pins can be directly set high or low using either the ″SET [m].i″ or ″CLR [m].i″ instructions respectively. The feature removes the need for programmers to first read the 8-bit output port, manipulate the input data to ensure that other bits are not changed and then output the port with the correct new data. This read-modify-write process is taken care of automatically when these bit operation instructions are used.

Table Read Operations

Data storage is normally implemented by using registers. However, when working with large amounts of fixed data, the volume involved often makes it inconvenient to store the fixed data in the Data Memory. To overcome this problem, Holtek microcontrollers allow an area of Program Memory to be setup as a table where data can be directly stored. A set of easy to use instructions provides the means by which this fixed data can be referenced and retrieved from the Program Memory.

Other Operations

In addition to the above functional instructions, a range of other instructions also exist such as the ″HALT″ instruction for Power-down operations and instructions to control the operation of the Watchdog Timer for reliable program operations under extreme electric or electromagnetic environments. For their relevant operations, refer to the functional related sections.

Instruction Set Summary

The following table depicts a summary of the instruction set categorised according to function and can be consulted as a basic instruction reference using the following listed conventions.

Table conventions

x: Bits immediate data
m: Data Memory address
A: Accumulator
i: 0~7 number of bits
addr: Program memory address

Mnemonic	Description	Cycles	Flag Affected
Arithmetic			
ADD A,[m]	Add Data Memory to ACC	1	Z, C, AC, OV
ADDM A,[m]	Add ACC to Data Memory	1Note	Z, C, AC, OV
ADD A,x	Add immediate data to ACC	1	Z, C, AC, OV
ADC A,[m]	Add Data Memory to ACC with Carry	1	Z, C, AC, OV
ADCM A,[m]	Add ACC to Data memory with Carry	1Note	Z, C, AC, OV
SUB A,x	Subtract immediate data from the ACC	1	Z, C, AC, OV
SUB A,[m]	Subtract Data Memory from ACC	1	Z, C, AC, OV
SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory	1Note	Z, C, AC, OV
SBC A,[m]	Subtract Data Memory from ACC with Carry	1	Z, C, AC, OV
SBCM A,[m]	Subtract Data Memory from ACC with Carry, result in Data Memory	1Note	Z, C, AC, OV
DAA [m]	Decimal adjust ACC for Addition with result in Data Memory	1Note	C
Logic Operation			
AND A,[m]	Logical AND Data Memory to ACC	1	Z
OR A,[m]	Logical OR Data Memory to ACC	1	Z
XOR A,[m]	Logical XOR Data Memory to ACC	1	Z
ANDM A,[m]	Logical AND ACC to Data Memory	1Note	Z
ORM A,[m]	Logical OR ACC to Data Memory	1Note	Z
XORM A,[m]	Logical XOR ACC to Data Memory	1Note	Z
AND A,x	Logical AND immediate Data to ACC	1	Z
OR A,x	Logical OR immediate Data to ACC	1	Z
XOR A,x	Logical XOR immediate Data to ACC	1	Z
CPL [m]	Complement Data Memory	1Note	Z
CPLA [m]	Complement Data Memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment Data Memory with result in ACC	1	Z
INC [m]	Increment Data Memory	1Note	Z
DECA [m]	Decrement Data Memory with result in ACC	1	Z
DEC [m]	Decrement Data Memory	1Note	Z
Rotate			
RRA [m]	Rotate Data Memory right with result in ACC	1	None
RR [m]	Rotate Data Memory right	1Note	None
RRCA [m]	Rotate Data Memory right through Carry with result in ACC	1	C
RRC [m]	Rotate Data Memory right through Carry	1Note	C
RLA [m]	Rotate Data Memory left with result in ACC	1	None
RL [m]	Rotate Data Memory left	1Note	None
RLCA [m]	Rotate Data Memory left through Carry with result in ACC	1	C
RLC [m]	Rotate Data Memory left through Carry	1Note	C

Mnemonic	Description	Cycles	Flag Affected
Data Move			
MOV A,[m]	Move Data Memory to ACC	1	None
MOV [m],A	Move ACC to Data Memory	1^Note	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of Data Memory	1^Note	None
SET [m].i	Set bit of Data Memory	1^Note	None
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if Data Memory is zero	1^Note	None
SZA [m]	Skip if Data Memory is zero with data movement to ACC	1^Note	None
SZ [m].i	Skip if bit i of Data Memory is zero	1^Note	None
SNZ [m].i	Skip if bit i of Data Memory is not zero	1^Note	None
SIZ [m]	Skip if increment Data Memory is zero	1^Note	None
SDZ [m]	Skip if decrement Data Memory is zero	1^Note	None
SIZA [m]	Skip if increment Data Memory is zero with result in ACC	1^Note	None
SDZA [m]	Skip if decrement Data Memory is zero with result in ACC	1^Note	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read			
TABRD [m]	Read table (specific page) to TBLH and Data Memory	2^Note	None
TABRDC [m]	Read table (current page) to TBLH and Data Memory	2^Note	None
TABRDL [m]	Read table (last page) to TBLH and Data Memory	2^Note	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear Data Memory	1^Note	None
SET [m]	Set Data Memory	1^Note	None
CLR WDT	Clear Watchdog Timer	1	TO, PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO, PDF
CLR WDT2	Pre-clear Watchdog Timer	1	TO, PDF
SWAP [m]	Swap nibbles of Data Memory	1^Note	None
SWAPA [m]	Swap nibbles of Data Memory with result in ACC	1	None
HALT	Enter power down mode	1	TO, PDF

Note: 1. For skip instructions, if the result of the comparison involves a skip then two cycles are required, if no skip takes place only one cycle is required.

2. Any instruction which changes the contents of the PCL will also require 2 cycles for execution.

3. For the "CLR WDT1" and "CLR WDT2" instructions the TO and PDF flags may be affected by the execution status. The TO and PDF flags are cleared after both "CLR WDT1" and "CLR WDT2" instructions are consecutively executed. Otherwise the TO and PDF flags remain unchanged.

Instruction Definition

ADC A,[m]	Add Data Memory to ACC with Carry
Description	The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the Accumulator.
Operation	ACC ← ACC + [m] + C
Affected flag(s)	OV, Z, AC, C

ADCM A,[m]	Add ACC to Data Memory with Carry
Description	The contents of the specified Data Memory, Accumulator and the carry flag are added. The result is stored in the specified Data Memory.
Operation	[m] ← ACC + [m] + C
Affected flag(s)	OV, Z, AC, C

ADD A,[m]	Add Data Memory to ACC
Description	The contents of the specified Data Memory and the Accumulator are added. The result is stored in the Accumulator.
Operation	ACC ← ACC + [m]
Affected flag(s)	OV, Z, AC, C

ADD A,x	Add immediate data to ACC
Description	The contents of the Accumulator and the specified immediate data are added. The result is stored in the Accumulator.
Operation	ACC ← ACC + x
Affected flag(s)	OV, Z, AC, C

ADDM A,[m]	Add ACC to Data Memory
Description	The contents of the specified Data Memory and the Accumulator are added. The result is stored in the specified Data Memory.
Operation	[m] ← ACC + [m]
Affected flag(s)	OV, Z, AC, C

AND A,[m]	Logical AND Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical AND operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″AND″ [m]
Affected flag(s)	Z

AND A,x	Logical AND immediate data to ACC
Description	Data in the Accumulator and the specified immediate data perform a bit wise logical AND operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″AND″ x
Affected flag(s)	Z

ANDM A,[m]	Logical AND ACC to Data Memory
Description	Data in the specified Data Memory and the Accumulator perform a bitwise logical AND operation. The result is stored in the Data Memory.
Operation	[m] ← ACC ″AND″ [m]
Affected flag(s)	Z

CALL addr	Subroutine call
Description	Unconditionally calls a subroutine at the specified address. The Program Counter then increments by 1 to obtain the address of the next instruction which is then pushed onto the stack. The specified address is then loaded and the program continues execution from this new address. As this instruction requires an additional operation, it is a two cycle instruction.
Operation	Stack ← Program Counter + 1 Program Counter ← addr
Affected flag(s)	None

CLR [m]	Clear Data Memory
Description	Each bit of the specified Data Memory is cleared to 0.
Operation	[m] ← 00H
Affected flag(s)	None

CLR [m].i	Clear bit of Data Memory
Description	Bit i of the specified Data Memory is cleared to 0.
Operation	[m].i ← 0
Affected flag(s)	None

CLR WDT	Clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared.

Operation	WDT cleared
	TO ← 0
	PDF ← 0

Affected flag(s)	TO, PDF

CLR WDT1	Pre-clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT2 and must be executed alternately with CLR WDT2 to have effect. Repetitively executing this instruction without alternately executing CLR WDT2 will have no effect.

Operation	WDT cleared
	TO ← 0
	PDF ← 0

Affected flag(s)	TO, PDF

CLR WDT2	Pre-clear Watchdog Timer
Description	The TO, PDF flags and the WDT are all cleared. Note that this instruction works in conjunction with CLR WDT1 and must be executed alternately with CLR WDT1 to have effect. Repetitively executing this instruction without alternately executing CLR WDT1 will have no effect.

Operation	WDT cleared
	TO ← 0
	PDF ← 0

Affected flag(s)	TO, PDF

CPL [m]	Complement Data Memory
Description	Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa.
Operation	[m] ← $\overline{[m]}$
Affected flag(s)	Z

CPLA [m]	Complement Data Memory with result in ACC
Description	Each bit of the specified Data Memory is logically complemented (1′s complement). Bits which previously contained a 1 are changed to 0 and vice versa. The complemented result is stored in the Accumulator and the contents of the Data Memory remain unchanged.
Operation	ACC ← $\overline{[m]}$
Affected flag(s)	Z

DAA [m]	Decimal-Adjust ACC for addition with result in Data Memory
Description	Convert the contents of the Accumulator value to a BCD (Binary Coded Decimal) value resulting from the previous addition of two BCD variables. If the low nibble is greater than 9 or if AC flag is set, then a value of 6 will be added to the low nibble. Otherwise the low nibble remains unchanged. If the high nibble is greater than 9 or if the C flag is set, then a value of 6 will be added to the high nibble. Essentially, the decimal conversion is performed by adding 00H, 06H, 60H or 66H depending on the Accumulator and flag conditions. Only the C flag may be affected by this instruction which indicates that if the original BCD sum is greater than 100, it allows multiple precision decimal addition.
Operation	[m] ← ACC + 00H or [m] ← ACC + 06H or [m] ← ACC + 60H or [m] ← ACC + 66H
Affected flag(s)	C

DEC [m]	Decrement Data Memory
Description	Data in the specified Data Memory is decremented by 1.
Operation	[m] ← [m] − 1
Affected flag(s)	Z

DECA [m]	Decrement Data Memory with result in ACC
Description	Data in the specified Data Memory is decremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC ← [m] − 1
Affected flag(s)	Z

HALT	Enter power down mode
Description	This instruction stops the program execution and turns off the system clock. The contents of the Data Memory and registers are retained. The WDT and prescaler are cleared. The power down flag PDF is set and the WDT time-out flag TO is cleared.

Operation	TO ← 0
	PDF ← 1

Affected flag(s)	TO, PDF

INC [m]	Increment Data Memory
Description	Data in the specified Data Memory is incremented by 1.
Operation	[m] ← [m] + 1
Affected flag(s)	Z

INCA [m]	Increment Data Memory with result in ACC
Description	Data in the specified Data Memory is incremented by 1. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC ← [m] + 1
Affected flag(s)	Z

JMP addr	Jump unconditionally
Description	The contents of the Program Counter are replaced with the specified address. Program execution then continues from this new address. As this requires the insertion of a dummy instruction while the new address is loaded, it is a two cycle instruction.
Operation	Program Counter ← addr
Affected flag(s)	None

MOV A,[m]	Move Data Memory to ACC
Description	The contents of the specified Data Memory are copied to the Accumulator.
Operation	ACC ← [m]
Affected flag(s)	None

MOV A,x	Move immediate data to ACC
Description	The immediate data specified is loaded into the Accumulator.
Operation	ACC ← x
Affected flag(s)	None

MOV [m],A	Move ACC to Data Memory
Description	The contents of the Accumulator are copied to the specified Data Memory.
Operation	[m] ← ACC
Affected flag(s)	None

NOP	No operation
Description	No operation is performed. Execution continues with the next instruction.
Operation	No operation
Affected flag(s)	None

OR A,[m]	Logical OR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical OR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″OR″ [m]
Affected flag(s)	Z

OR A,x	Logical OR immediate data to ACC
Description	Data in the Accumulator and the specified immediate data perform a bitwise logical OR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″OR″ x
Affected flag(s)	Z

ORM A,[m]	Logical OR ACC to Data Memory
Description	Data in the specified Data Memory and the Accumulator perform a bitwise logical OR operation. The result is stored in the Data Memory.
Operation	[m] ← ACC ″OR″ [m]
Affected flag(s)	Z

RET	Return from subroutine
Description	The Program Counter is restored from the stack. Program execution continues at the restored address.
Operation	Program Counter ← Stack
Affected flag(s)	None

RET A,x	Return from subroutine and load immediate data to ACC
Description	The Program Counter is restored from the stack and the Accumulator loaded with the specified immediate data. Program execution continues at the restored address.
Operation	Program Counter ← Stack ACC ← x
Affected flag(s)	None

RETI	Return from interrupt
Description	The Program Counter is restored from the stack and the interrupts are re-enabled by setting the EMI bit. EMI is the master interrupt global enable bit. If an interrupt was pending when the RETI instruction is executed, the pending Interrupt routine will be processed before returning to the main program.
Operation	Program Counter ← Stack EMI ← 1
Affected flag(s)	None

RL [m]	Rotate Data Memory left
Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0.
Operation	[m].(i+1) ← [m].i; (i=0~6) [m].0 ← [m].7
Affected flag(s)	None

RLA [m]	Rotate Data Memory left with result in ACC
Description	The contents of the specified Data Memory are rotated left by 1 bit with bit 7 rotated into bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation	ACC.(i+1) ← [m].i; (i=0~6)
	ACC.0 ← [m].7

Affected flag(s)	None

RLC [m]	Rotate Data Memory left through Carry
Description	The contents of the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into bit 0.
Operation	[m].(i+1) ← [m].i; (i=0~6) [m].0 ← C C ← [m].7
Affected flag(s)	C

RLCA [m]	Rotate Data Memory left through Carry with result in ACC
Description	Data in the specified Data Memory and the carry flag are rotated left by 1 bit. Bit 7 replaces the Carry bit and the original carry flag is rotated into the bit 0. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation	ACC.(i+1) ← [m].i; (i=0~6)
	ACC.0 ← C
	C ← [m].7

Affected flag(s)	C

RR [m]	Rotate Data Memory right
Description	The contents of the specified Data Memory are rotated right by 1 bit with bit 0 rotated into bit 7.
Operation	[m].i ← [m].(i+1); (i=0~6) [m].7 ← [m].0
Affected flag(s)	None

RRA [m]	Rotate Data Memory right with result in ACC
Description	Data in the specified Data Memory and the carry flag are rotated right by 1 bit with bit 0 rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation	ACC.i ← [m].(i+1); (i=0~6)
	ACC.7 ← [m].0

Affected flag(s)	None

RRC [m]	Rotate Data Memory right through Carry
Description	The contents of the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7.
Operation	[m].i ← [m].(i+1); (i=0~6) [m].7 ← C C ← [m].0
Affected flag(s)	C

RRCA [m]	Rotate Data Memory right through Carry with result in ACC
Description	Data in the specified Data Memory and the carry flag are rotated right by 1 bit. Bit 0 replaces the Carry bit and the original carry flag is rotated into bit 7. The rotated result is stored in the Accumulator and the contents of the Data Memory remain unchanged.

Operation	ACC.i ← [m].(i+1); (i=0~6)
	ACC.7 ← C
	C ← [m].0

Affected flag(s)	C

SBC A,[m]	Subtract Data Memory from ACC with Carry
Description	The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	ACC ← ACC − [m] − $\overline{C}$
Affected flag(s)	OV, Z, AC, C

SBCM A,[m]	Subtract Data Memory from ACC with Carry and result in Data Memory
Description	The contents of the specified Data Memory and the complement of the carry flag are subtracted from the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	[m] ← ACC − [m] − $\overline{C}$
Affected flag(s)	OV, Z, AC, C

SDZ [m]	Skip if decrement Data Memory is 0
Description	The contents of the specified Data Memory are first decremented by 1. If the result is 0 the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	[m] ← [m] − 1 Skip if [m]=0
Affected flag(s)	None

SDZA [m]	Skip if decrement Data Memory is zero with result in ACC
Description	The contents of the specified Data Memory are first decremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.
Operation	ACC ← [m] − 1 Skip if ACC=0
Affected flag(s)	None

SET [m]	Set Data Memory
Description	Each bit of the specified Data Memory is set to 1.
Operation	[m] ← FFH
Affected flag(s)	None

SET [m].i	Set bit of Data Memory
Description	Bit i of the specified Data Memory is set to 1.
Operation	[m].i ← 1
Affected flag(s)	None

SIZ [m]	Skip if increment Data Memory is 0
Description	The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	[m] ← [m] + 1 Skip if [m]=0
Affected flag(s)	None

SIZA [m]	Skip if increment Data Memory is zero with result in ACC
Description	The contents of the specified Data Memory are first incremented by 1. If the result is 0, the following instruction is skipped. The result is stored in the Accumulator but the specified Data Memory contents remain unchanged. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	ACC ← [m] + 1 Skip if ACC=0
Affected flag(s)	None

SNZ [m].i	Skip if bit i of Data Memory is not 0
Description	If bit i of the specified Data Memory is not 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is 0 the program proceeds with the following instruction.
Operation	Skip if [m].i ≠ 0
Affected flag(s)	None

SUB A,[m]	Subtract Data Memory from ACC
Description	The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	ACC ← ACC − [m]
Affected flag(s)	OV, Z, AC, C

SUBM A,[m]	Subtract Data Memory from ACC with result in Data Memory
Description	The specified Data Memory is subtracted from the contents of the Accumulator. The result is stored in the Data Memory. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	[m] ← ACC − [m]
Affected flag(s)	OV, Z, AC, C

SUB A,x	Subtract immediate data from ACC
Description	The immediate data specified by the code is subtracted from the contents of the Accumulator. The result is stored in the Accumulator. Note that if the result of subtraction is negative, the C flag will be cleared to 0, otherwise if the result is positive or zero, the C flag will be set to 1.
Operation	ACC ← ACC − x
Affected flag(s)	OV, Z, AC, C

SWAP [m]	Swap nibbles of Data Memory
Description	The low-order and high-order nibbles of the specified Data Memory are interchanged.
Operation	[m].3~[m].0 ↔ [m].7~[m].4
Affected flag(s)	None

SWAPA [m]	Swap nibbles of Data Memory with result in ACC
Description	The low-order and high-order nibbles of the specified Data Memory are interchanged. The result is stored in the Accumulator. The contents of the Data Memory remain unchanged.
Operation	ACC.3~ACC.0 ← [m].7~[m].4 ACC.7~ACC.4 ← [m].3~[m].0
Affected flag(s)	None

SZ [m]	Skip if Data Memory is 0
Description	If the contents of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	Skip if [m]=0
Affected flag(s)	None

SZA [m]	Skip if Data Memory is 0 with data movement to ACC
Description	The contents of the specified Data Memory are copied to the Accumulator. If the value is zero, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0 the program proceeds with the following instruction.
Operation	ACC ← [m] Skip if [m]=0
Affected flag(s)	None

SZ [m].i	Skip if bit i of Data Memory is 0
Description	If bit i of the specified Data Memory is 0, the following instruction is skipped. As this requires the insertion of a dummy instruction while the next instruction is fetched, it is a two cycle instruction. If the result is not 0, the program proceeds with the following instruction.
Operation	Skip if [m].i=0
Affected flag(s)	None

TABRD [m]	Read table (specific page) to TBLH and Data Memory
Description	The low byte of the program code addressed by the table pointer pair (TBHP and TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None

TABRDL [m]	Read table (current page) to TBLH and Data Memory
Description	The low byte of the program code (current page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None

TABRDL [m]	Read table (last page) to TBLH and Data Memory
Description	The low byte of the program code (last page) addressed by the table pointer (TBLP) is moved to the specified Data Memory and the high byte moved to TBLH.
Operation	[m] ← program code (low byte) TBLH ← program code (high byte)
Affected flag(s)	None

XOR A,[m]	Logical XOR Data Memory to ACC
Description	Data in the Accumulator and the specified Data Memory perform a bitwise logical XOR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″XOR″ [m]
Affected flag(s)	Z

XORM A,[m]	Logical XOR ACC to Data Memory
Description	Data in the specified Data Memory and the Accumulator perform a bitwise logical XOR operation. The result is stored in the Data Memory.
Operation	[m] ← ACC ″XOR″ [m]
Affected flag(s)	Z

XOR A,x	Logical XOR immediate data to ACC
Description	Data in the Accumulator and the specified immediate data perform a bitwise logical XOR operation. The result is stored in the Accumulator.
Operation	ACC ← ACC ″XOR″ x
Affected flag(s)	Z

Package Information

Note that the package information provided here is for consultation purposes only. As this information may be updated at regular intervals users are reminded to consult the Holtek website for the latest version of the Package/Carton Information.

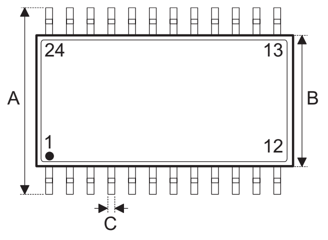
Additional supplementary information with regard to packaging is listed below. Click on the relevant section to be transferred to the relevant website page.

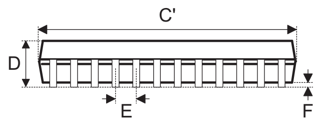
- Package Information (include Outline Dimensions, Product Tape and Reel Specifications)

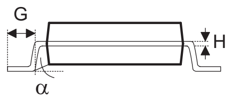
- The Operation Instruction of Packing Materials

- Carton information

24-pin SOP (300mil) Outline Dimensions



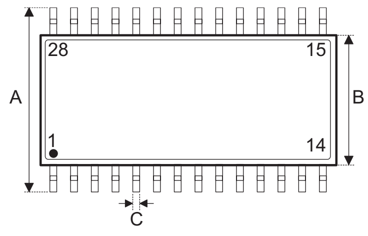


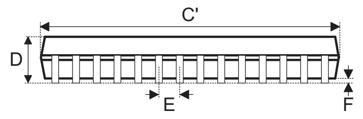


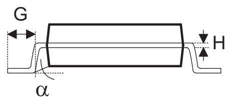
Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.406 BSC	—
B	—	0.295 BSC	—
C	0.012	—	0.020
C’	—	0.606 BSC	—
D	—	—	0.104
E	—	0.050 BSC	—
F	0.004	—	0.012
G	0.016	—	0.050
H	0.008	—	0.013
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	10.30 BSC	—
B	—	7.5 BSC	—
C	0.31	—	0.51
C’	—	15.4 BSC	—
D	—	—	2.65
E	—	1.27 BSC	—
F	0.10	—	0.30
G	0.40	—	1.27
H	0.20	—	0.33
α	0°	—	8°

28-pin SOP (300mil) Outline Dimensions







Symbol	Dimensions in inch		
	Min.	Nom.	Max.
A	—	0.406 BSC	—
B	—	0.295 BSC	—
C	0.012	—	0.020
C’	—	0.705 BSC	—
D	—	—	0.104
E	—	0.050 BSC	—
F	0.004	—	0.012
G	0.016	—	0.050
H	0.008	—	0.013
α	0°	—	8°

Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	10.30 BSC	—
B	—	7.5 BSC	—
C	0.31	—	0.51
C’	—	17.9 BSC	—
D	—	—	2.65
E	—	1.27 BSC	—
F	0.10	—	0.30
G	0.40	—	1.27
H	0.20	—	0.33
α	0°	—	8°